平成2年9月1日発行(毎月1回発行) 第1巻 第2号 通巻2号

コンピュータ技術総合誌 月刊 スーパーアスキー

Super ASCII

FOR ADVANCED COMPUTER USERS

9

創刊 第2号

INSIDE SOFTWARE

32bit対応に拡張された

OS/2 Ver. 2.0 のアーキテクチャ

TEST LAB.

Windows Ver. 3.0対応の
ハイパーテキストツール

ToolBook

Sunの低価格 ワークステーション

SPARCstation SLC

特別企画

超現実を現実とする仮想現実は,どこまで現実的か?

Virtual Reality

eyephone, dataglove, datasuit

特集

'90年代のOSが備えるべき機能を追究する

オペレーティングシステムとは何か

OSとMS-DOSはどこが違うか

SONY

河原博和
スーパーマイクロ事業本部
ワークステーション事業部
設計2課

夢から先に作りました。

Laptop NEWS NWS-1250 [NWS-1230]

●CPU:68030(25MHz)●コプロセッサー:68882(25MHz)●MIPS値:3.9●メインメモリー:8MB[4MB]/拡張最大12MB●3.5インチFDD:1.44MB●内蔵ハードディスク:240MB[200MB]●標準インターフェース:イーサネット、SCSI、RS-232C、パラレル●バックライト付き高解像度大型液晶ディスプレイ(1120×780ドット)●マルチメディア対応のスピーカー付きオーディオインターフェース内蔵●モデムボード(NWB-238)など周辺ボードが利用可能な専用I/Oスロット

※イーサネットはゼロックス社の登録商標です。
※UNIXはAT&Tおよびベル研究所が開発し、AT&Tがライセンスしています。

夢が、はじまり。

第3回懸賞論文募集

「21世紀のコンピューター：夢を現実に」

物語には、はじまりがあります。テクノロジーの進化にもスタートがあるとしたら、それはきっと、人の夢でしょう。たとえば、UNIX環境のワークステーションを家に持ち帰って、仕事の続きがしたい。そんな夢を、技術者のひとりが持たなかったら、フルスペックのワークステーションをラップトップに凝縮したNEWSは、決して生まれなかったのです。夢を、現実に。ソニーは、夢の大切さを知っています。

第3回懸賞論文募集の概要

◎論文テーマ：「21世紀のコンピューター：夢を実現に」◎枚数：A4判25枚程度 ◎賞品及び研究助成金：最優秀賞／NEWS（NWS-1250、拡張RAM 4MB）＋研究助成金50万円、優秀賞／NEWS（NWS-1250）＋研究助成金30万円、佳作／NEWS（NWS-1230）＋研究助成金10万円（以上各賞1チーム）、特別賞／PalmTop（3チーム） ◎応募締切：1991年3月31日 ◎審査委員長：相磯秀夫 ◎副審査委員長：田中英彦 ◎審査委員（50音順）：安西祐一郎、片山卓也、川合慧、岸田孝一、斎藤信男、竹内郁雄、田中譲、辻井潤一、徳田英幸、中島玲二、二木厚吉、村井純、米澤明憲、所真理雄 ◎主催：ソニー株式会社 ◎協賛：ソニーコンピュータサイエンス研究所、NEWSグループ

応募条件

応募論文は未発表で、応募者自身の構想／考案に基づくものに限定します。現在存在しない技術や素子の存在を想定して設計することは自由ですが、その際には仮定の仕様を明記してください。入選論文については著作権をソニー株式会社に譲渡していただき、さらに国立国会図書館に寄贈いたしますので、論文の内容は公知のものになります。著作に関連する発明／考案の工業所有権の取得を希望される場合は、投稿前にその旨お申し出ください。

第2回懸賞論文募集　受賞者

◎最優秀賞：熊沢逸夫「相互作用を及ぼしあう微生物集団を利用した計算について」◎優秀賞：丸一威雄「SimWorld：視覚、聴覚、触覚インターフェースと結合した仮想世界シミュレーション・モデル」◎佳作：加納慎一「パーソナルコンピュータを利用した、小学校における視聴覚室構想」◎特別賞（50音順）：小郷直言「双方向遠隔教育用ハイパメディアシステムの設計」、砂原秀樹「NextBook：可搬型ワークステーション実現に関する考察」、渕武志「脳の睡眠モードを利用したマン・マシンインターフェース」、和田勉「計算機教育用システムと『AV講義録』」

◎応募希望の方には募集要項をお送りします。ハガキに住所、氏名、年齢、電話番号、職業を明記のうえ〒141東京都品川区東五反田1-22-1 五反田ANビル ソニー株式会社「21世紀のコンピューター」懸賞論文募集係まで。

開放型分散処理ワークステーション

NEWS

ソニー株式会社 NEWSインフォメーションセンター

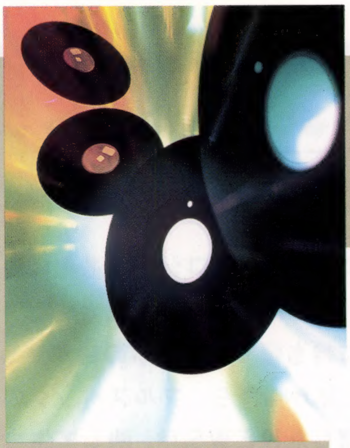
〒141 東京都品川区東五反田1-22-1 五反田ANビル　TEL(03)448-3553

懸賞論文募集によせて

「21世紀のコンピューター：夢を現実に」と題し、若手研究者、学生の夢を募る懸賞論文募集も、今年で3回目を迎えることになりました。昨年の第2回懸賞論文募集ではまだまだ論文数こそ少数ですが、内容としては充実したものが増えてきたように思います。　夢なくしては、技術の進歩もありません。なるべく大きなスケールで、独創性を思う存分に発揮して、夢を描いてください。昨年、一昨年を超える、活気あふれた論文を期待しています。

慶應義塾大学　環境情報学部長　相磯秀夫

資料請求 No.001



photo/Susumu Uchino

Super ASCII

September
1990 No.2

33 Super EXPRESS

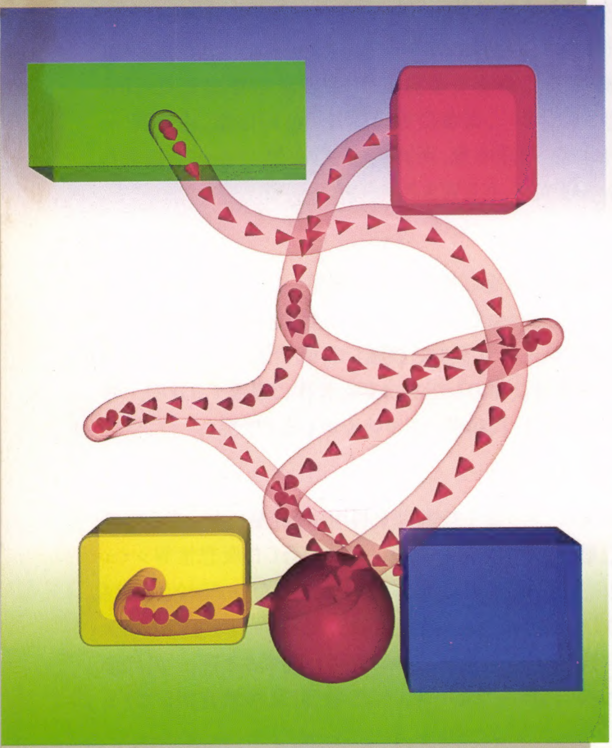
SPARCstation IPC／低価格NEC PostScriptプリンタ／USENIXレポート／速報Digital World／イベントカレンダー

57 特集 オペレーティングシステムとは何か

OSとMS-DOSはどこが違うか

MS-DOSの持っていない機能とは何であるかを出発点にして，オペレーティングシステムが基本的に持つべき機能と構造を追究する．さらにアプリケーションの新しい機能を実現するために，どのような新しいOSが必要になるのかを探る．

TSRプログラムの限界／排他制御の必要性／フラグメンテーションする640K／EXE, COM, TXT, BATしかいらない／新しい機能／debate



81 INSIDE SOFTWARE

32bit環境に対応したOS/2 Ver.2.0

現在開発中のOS/2 Ver.2.0は，386ネイティブの機能を活用し，Ver.1.2からさまざまな拡張が施されている．今回は特に，そのメモリ管理機構と開発環境について紹介する．

97 TEST LAB.

98 ToolBook
ハイパーテキストで綴る本の世界

104 SPARCstation SLC
ディスクレスで低価格それでいて高性能

110 QMS ColorScript 100 Model-10J
日本語PostScript搭載のカラープリンタ

114 Hardware＝Software＋Hardware

佐倉はじめ

●Non StopコンピュータのためのGUARDIAN90

118 特別企画 Virtual Reality

武邑光裕／伊藤カビン

●超現実を現実とする仮想現実は，どこまで現実的か？
Interview：Jaron Lanier, Rudy Rucker

CONTENTS

連載

125 システム探検隊 ── ●国立天文台の巻　船田　巧

128 Astro Guide '90 ──

129 プロフェッショナルワークベンチとしてのUNIX入門 ──
第2回 EmacsとWnn　村井　純

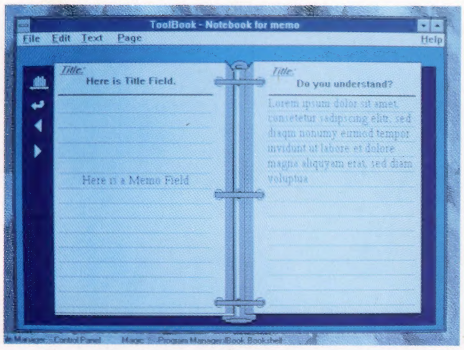
135 OS/2アーキテクチャ入門 ── ●プログラムの実行環境
佐藤和彦＋山田伸一郎

プログラミングフォーラム

142 386de MS-DOS ── リアルモードでdword その2
明石真平

148 入門システムプログラム ── 画面書き換えの最適化
小野芳彦

154 ウィンドウシステムのプログラミングパラダイム ──
中川幸男



161 EP news

162 基礎講座　文書交換のための共通文書フォーマットの動向
村上裕一

169 キーワード　文字，字体，字形，フォント(2) ──
小町祐史

174 トピックス　世界初の"Digital World"レポート ──

176 業界動向　コンピュータ化へ急シフトをする軽印刷

177 MIPS

178 パソコン1024夜 ── 福崎俊博

180 プログラマのための数学的世界観入門 ── 斎藤　明

182 ウサギと月と，そして太陽と ── 砂原秀樹

184 Fat Bits ── 山崎俊一

186 Book Shelf ──

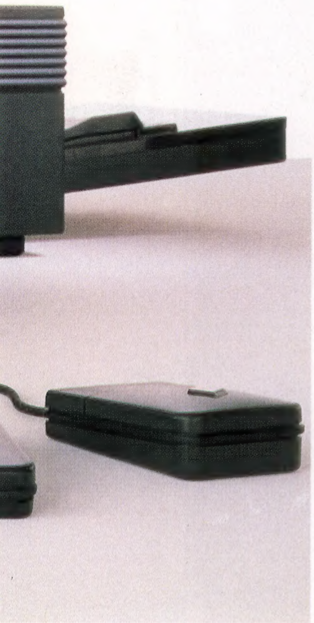
188 読者からのお便りコーナー ──

190 最新ソフトウェアバージョンリスト ──

資料請求 No.004

10年後。

このマシンには、10年後のコンピュータの姿が見える。
未来をひらく、7つのブレークスルーがある。
1990年代のスーパーワークステーション、NeXT。
キヤノンから。

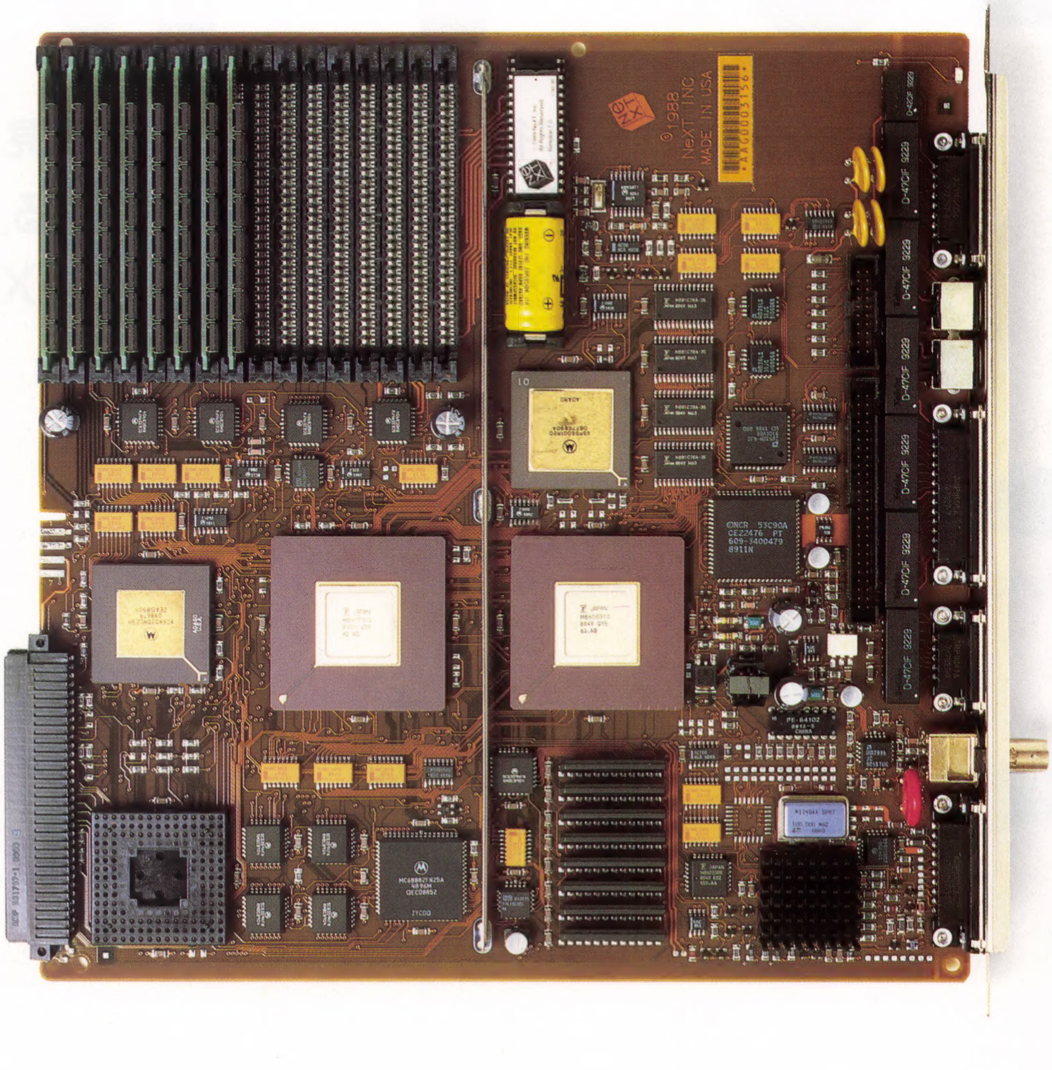


NeXT™

Canon

キヤノン株式会社 〒211 神奈川県川崎市幸区鹿島田890-12 新川崎三井ビル TEL.(044)549-5295 キヤノンソフトウェア株式会社 〒108 東京都港区三田3-9-7 TEL.(03)455-9920

NeXT発。コンピュータの未来をひらく、7つのブレークスルー①



超高速スループットをデスクトップサイズで。
NeXTは、メインフレームを
1枚のボードに凝縮しました。

1990年代のコンピュータNeXTは、超高速スループットを追求しました。

プロセッサの速さだけで、コンピュータの処理速度を判断することはできません。チップ単体のスピードではなく、マシン全体のスピードが問題となります。これからの10年間、コンピュータの未来をひらくにふさわしい性能を考えるには、1秒間に処理できる情報量、スループットがたいへん重要です。1990年代のコンピュータに求められるスループットを達成するため、NeXTはメインフレームに注目しました。メインフレームは作業効率だけを追求し、マシンのサイズとかコストといった制約をはずしてつくられたコンピュータ。メモリ領域に直結して単独の入出力処理を行うプロセッサをいくつも備えています。デスクトップサイズのメインフレームをつくりあげること。それが、NeXTのチャレンジでした。

独自開発のVLSIチップの採用で、メインフレームを1枚のボードに。

メインフレームと同等の高度なアーキテクチャをデスクトップサイズに凝縮するため、NeXTは独自開発の2つのVLSIチップを採用しました。一方のチップには12個の入出力プロセッサが入っており、それぞれがメモリ領域に直結。もう一方のチップには、ぼう大なメモリを管理するための回路が入っています。結果として、メインフレームをわずか45個のチップからなる1フィート四方のシステムボードに納めました。さらに、CPU68030、FPU68882、デジタル信号プロセッサ(DSP)56001を搭載。いずれのプロセッサもクロック25MHzのハイスピードで動作し、32MB/secという、これまでのデスクトップサイズのコンピュータの常識を完全に打ち破る超高速スループットを実現しました。

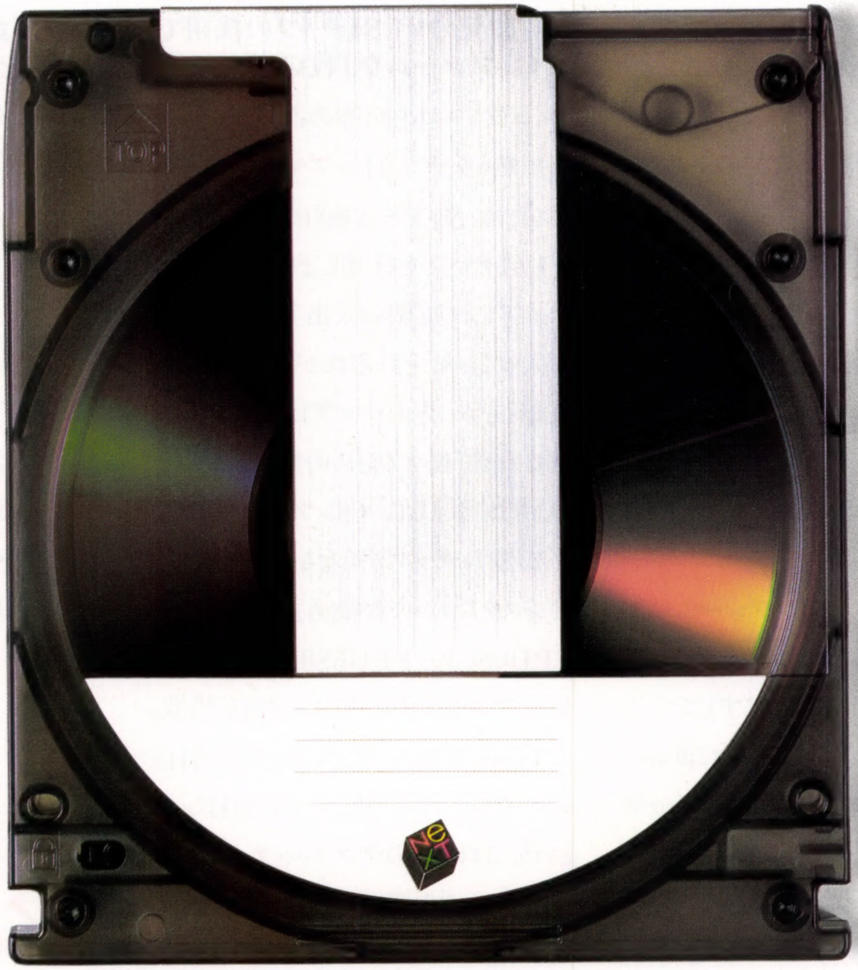
NeXTは、速いだけではありません。その広大な拡張性も、未来のために。

システムボードは、4つあるNeXT本体のスロットのひとつに装着されています。他の3つのスロットは、1990年代といっしょに進化していくために。このスロットにプラグインするだけで、ギガバイトレベルのメモリやより高度な分散処理などの新技術を将来内蔵していくことが可能です。NeXTは、ただ単に速いだけのマシンではありません。コンピュータの未来を積極的に取り込んでいくことができる大きな「器」も併せ持っています。

NeXT™

Canon

NeXT発。コンピュータの未来をひらく、7つのブレークスルー②



読み書きも、持ち運びもできる
大容量256MBの光磁気ディスク。
NeXTの場合、記憶メディアも1990年代です。

NeXTが選んだ、1990年代の記憶メディア。それが光磁気ディスクです。

1970年代、情報はフロッピーディスクに記録されました。その特性は、安価で、信頼性が高く、持ち運びもできること。1980年代になると、新しい記憶メディアが導入されました。ハードディスクです。携帯には不向きですが、大容量記憶と高速アクセスを達成しました。そして1990年代。NeXTは、これからの10年間を担うにふさわしい、新次元の記憶メディアを追求しました。それは大容量で信頼性が高く、携帯可能、またコストパフォーマンスに優れていることが必要です。外部記憶装置として、NeXTは光磁気ディスクドライブを採用しました。光磁気ディスクは、持ち運びできる大容量256MB。リード/ライト/消去が可能で、経時劣化の面でも優れた信頼性を実現した1990年代の記憶メディアです。

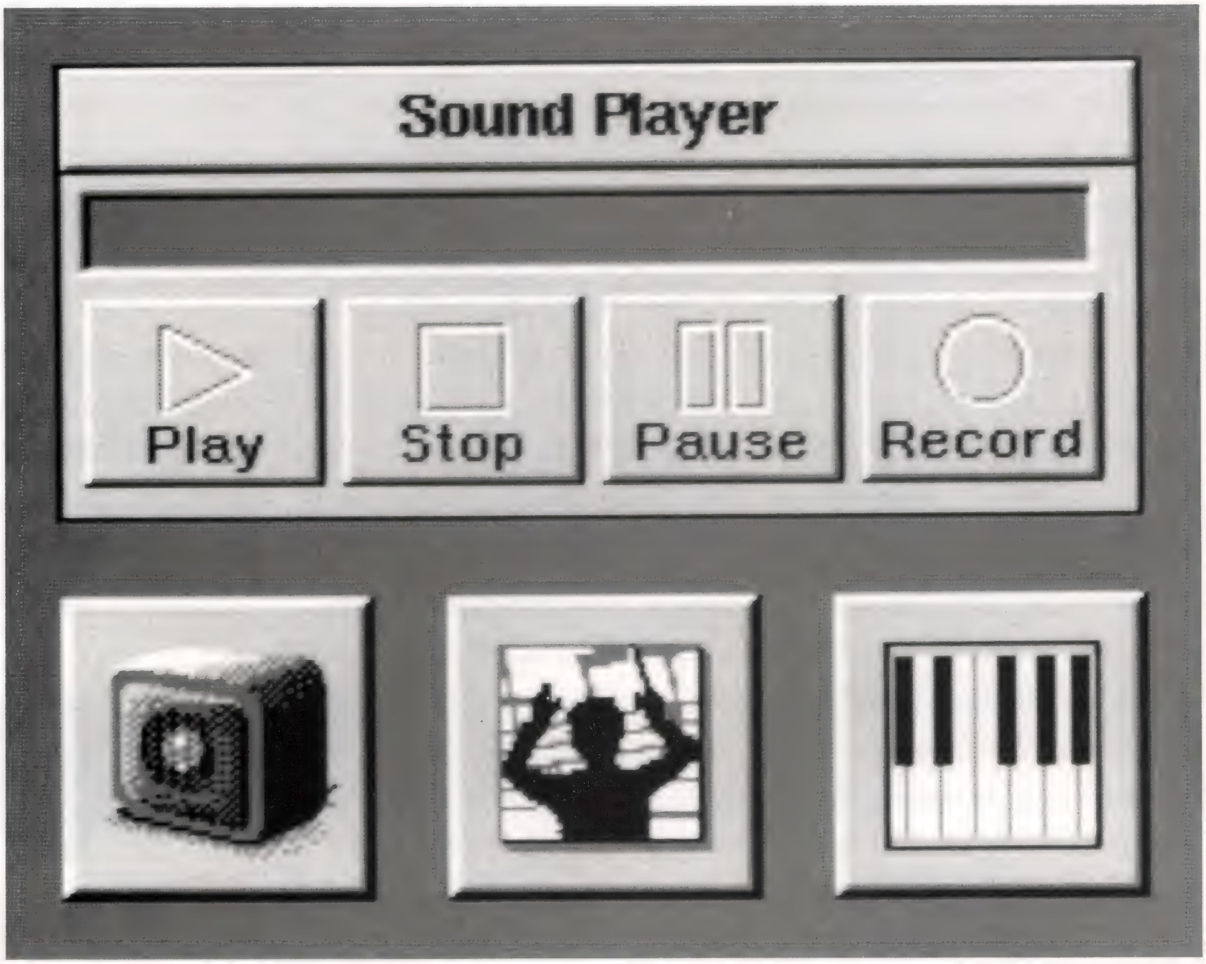
最新のレーザ技術が、優れた信頼性の大容量記憶メディアを実現しました。

NeXTの光磁気ディスクドライブは、毎分3,000回転し、データの読み書きをいずれも1本のレーザによって行います。デジタル情報を忠実にキープするために、データを0または1のいずれかで記憶。読み出しは、低出力のレーザをディスクに照射してその反射を読み取り、0または1として判定します。一方、データの書き込みは、まず消去から。レーザで目的のディスク部分を加熱し、さらに電磁装置でそれぞれの結晶を0の位置に設定。書き込みの段階では、必要な結晶だけを再びレーザで1にセットします。このように光磁気ディスクドライブは機械的なヘッドを持っていないので、メディアが破壊される心配がありません。また、消去しても経時劣化がないので、何度でも繰り返し読み書きすることができます。

大容量256MBの光磁気ディスクが、大量のバンドルソフトをNeXTへ。

光磁気ディスクは1枚で256MB。この大容量記憶が、NeXTへの大量なソフトウェアのバンドルをサポートしています。MACHなどのシステムファイルをはじめワードプロセッシングやDTPソフト、Digital Websterなどの辞書類、シェークスピア全集を含むDigital Librarian、Objective C、各種デモプログラムなど、すべてのソフトウェアが1枚のディスクに。光磁気ディスクといっしょに、NeXTは、ソフトウェアの「大陸」を提供しています。

Canon


Sound Player
Play
Stop
Pause
Record

DSPチップ標準搭載で、CDと同等の高音質。
NeXTのサウンドが
1990年代のスタンダードになります。

グラフィックスからサウンドへ。1990年代、コンピュータの世界が拡大します。

コンピュータグラフィックスという考え方は、かつて途方もないことだと思われていました。しかし今日では、文字、数字にくわえてグラフィックスをコンピュータで処理することが当たり前になっています。現在、サウンドがひと昔前のグラフィックスと同じようなポジションにあるのかもしれません。サウンドは、これまでコンピュータにとってあまり必要のないものと考えられてきました。ところが、コンピュータの使用領域が広がるにつれて、サウンドの持つ多様なアプリケーションの可能性が注目されるようになってきたのです。サウンドを無視したコンピュータは、まったく時代遅れのものになってしまうでしょう。1990年代のコンピュータシーンを想定し、NeXTはいちはやく高度なサウンド機能を取り入れました。

NeXTは、DSPチップを標準搭載。サウンド処理を、CD並の高音質で。

サウンドを自由自在に操るためには、高度なマトリクス処理能力が必要とされます。NeXTは、CPU68030、FPU68882とともに、コンピュータで初めてデジタル信号プロセッサ(DSP)56001を標準搭載しました。このDSPチップが、大量のマトリクス計算を25MHzの高速で実行。44.1KHzのサンプリングレート、16ビットの分解能、そして完全ステレオという、CDと同じクオリティのサウンドを作成することができます。また、特定の目的のために、DSPチップそのものをプログラムすることも可能です。たとえば、ファックスや高速モデムの機能。さらには、音声合成や音声認識の機能などを構築することができるのです。ある日、NeXTが口頭の命令を理解するようになるかもしれません。

高音質のサウンド処理が、さまざまなアプリケーションを可能にしました。

NeXTのサウンド機能は、さまざまな可能性を持っています。ひとつの例として、テキストとグラフィックスだけでなく、サウンドや音声を組み合わせた電子メール。あるいは、ビジュアルによるフィードバックと同時にサウンドによるフィードバックを行う科学的なシミュレーションモデル。さらに、NeXTはヘッドホン用ジャックとステレオRCAジャックを装備しているので、別のオーディオシステムと簡単に接続することができます。

NeXT発。コンピュータの未来をひらく、7つのブレークスルー④

ディスプレイ上のイメージをそのままプリンタに。NeXTが、CRT画面とプリントアウトをひとつにしました。

NeXTのプリントイメージシステムは世界初Display PostScriptです。

質の高いアウトプットを実現するために、現在では数多くの高性能プリンタがPostScriptを採用しています。このPostScriptによって、さまざまなフォントの利用が可能に。そしてテキストとグラフィックスを混在させることもできるようになりました。つまり、自由自在に加工したイメージを書き出すことができるようになったのです。しかし、これまではプリンタだけがPostScriptによってコントロールされ、ディスプレイ上のイメージはまったく別のシステムで処理されていました。NeXTは、従来のPostScriptをさらに発展させた新しいプリントイメージシステム、Display PostScriptを採用。ディスプレイとプリンタのイメージ処理を、世界で初めて、ひとつのシステムに統一しました。

新しいプリントイメージシステムが、高品位プリンティングを身近なものに。

NeXTのDisplay PostScriptは、イメージをPostScriptベースで直接CRT画面上につくりだします。拡大、縮小や回転をしても、画面上のイメージは鮮明で、その輪郭はなめらかです。プリントアウトの際は、NeXT本体でつくられたイメージがダイレクトに高精度400dpiレーザプリンタへ。高解像度のMegaPixel Display上のイメージ通りの高品質なアウトプットができます。また、PostScriptによる処理をプリンタ側で改めて行う必要がないので、印刷の指示から出力までの時間が短縮されました。さらに、プリンタ側に独自処理のためのCPUやメモリが不要となり、高性能レーザプリンタをよりコンパクトに、より低価格で提供することができるようになりました。

Display PostScriptは、1990年代のユーザーとプログラマーのために。

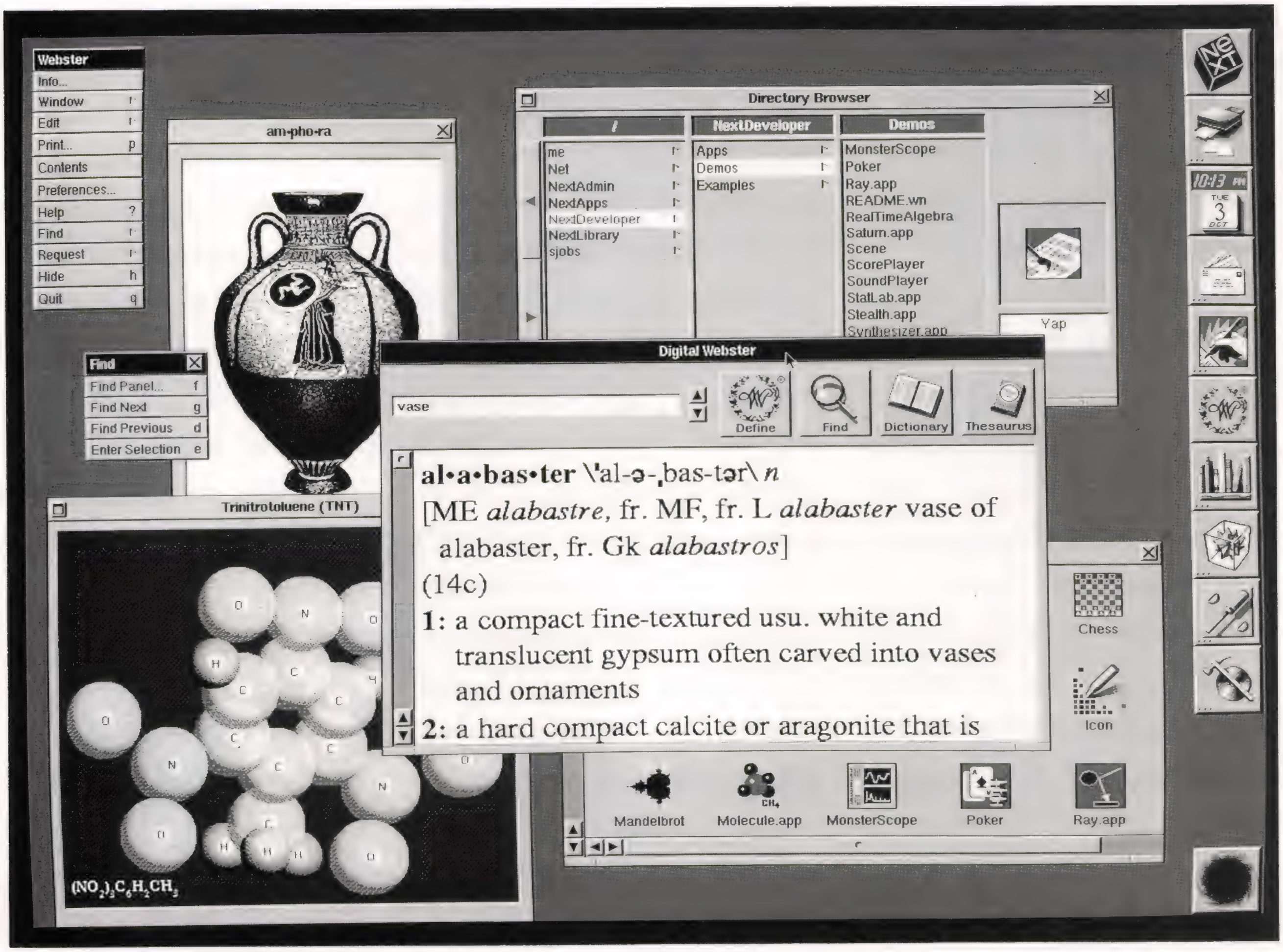
Display PostScriptは、互換性に優れた1990年代の標準システムです。ハードウェアから独立したシステムなので、Display PostScriptで定義されたイメージは、PostScriptベースであればどのプリンタでも同じアウトプットができます。また、ディスプレイとプリンタのイメージ処理プログラムをそれぞれ別個につくる必要がありません。ユーザーだけではなく、プログラマーにとっても大きなメリットがあります。

NeXT™

Canon

●Display PostScriptは米国アドビシステム社の商標です。PostScriptは米国アドビシステム社の登録商標です。

資料請求 No.013


Webster
Info...
Window
Edit
Print...
Contents
Preferences...
Help
Find
Request
Hide
Quit
am·pho·ra
Directory Browser
NextDeveloper
Demos
me
Net
NextAdmin
NextApps
NextDeveloper
NextLibrary
sjobs
Apps
Demos
Examples
MonsterScope
Poker
Ray.app
README.wn
RealTimeAlgebra
Saturn.app
Scene
ScorePlayer
SoundPlayer
StatLab.app
Stealth.app
Synthesizer.app
Yap
Find
Find Panel...
Find Next
Find Previous
Enter Selection
Digital Webster
vase
Define
Find
Dictionary
Thesaurus
al•a•bas•ter \'al-ə-ˌbas-tər\ n
[ME alabastre, fr. MF, fr. L alabaster vase of alabaster, fr. Gk alabastros]
(14c)
1: a compact fine-textured usu. white and translucent gypsum often carved into vases and ornaments
2: a hard compact calcite or aragonite that is
Trinitrotoluene (TNT)
$(NO_2)_3C_6H_2CH_3$
Chess
Icon
Mandelbrot
Molecule.app
MonsterScope
Poker
Ray.app
10:13 PM
TUE
3
OCT

高性能をもっと使いやすく。
NeXTは、UNIXコンピュータの顔つきをすっかり変えてしまいました。

1990年代のオペレーティングシステムに、NeXTはUNIXを選択しました。

マルチタスク処理とネットワーキング。これからのコンピュータに必要とされる2つの面で、UNIXはたいへん優れたオペレーティングシステムです。NeXTはカーネギ・メロン大学が開発した分散型OS、MACHを搭載しました。これはバークレイ版UNIX(4.3BSD)とコンパチブルで、共有メモリ、高速プロセス間通信、スレッドを使ったマルチタスクサポートの可能性などの点でさまざまな拡張がくわえられたものです。しかし、UNIXは、単純なコマンドでも非常に複雑で、その難しい操作環境が多くのユーザーをせっかくの高性能から遠ざけています。いかにUNIXのハイパワーを人にわかりやすく、使いやすくするか。NeXTの新しいグラフィックインタフェースが、その問題を解決しました。

●UNIXは米国AT&T社の登録商標です。

グラフィックインタフェースが、UNIXのハイパワーをすべての人の手に。

NeXTは、マウスで直感的、視覚的に操作できるユーザーフレンドリーなグラフィックインタフェースを採用しました。このインタフェースでは、簡単なマウス操作で複雑な機能にアクセスが可能。だれでもUNIXのハイパワーを、気軽に利用することができます。マルチウィンドウ上では、メモリの容量が許す限り何種類のプログラムでも同時に動かすことが可能になりました。ひとつのプログラム進行中に別のプログラムに移っても、最初のプログラムはそのまま処理を続けます。また、プロフェッショナルユーザーは、いままで使い慣れた方法でUNIXにアクセスすることもできます。コマンドウィンドウを呼び出し、UNIXとオリジナルの形で直接インタフェースすることもできるのです。

人にわかりやすい、NeXTのワークスペース。パーソナルに利用できます。

NeXTのワークスペースには、独自の新機能が装備されました。たとえば画面の右端には、利用頻度の高いアプリケーションプログラムに素早くアクセスするためのドックがあります。不要になったファイルは画面右下のブラックホールへ。さらに、メニューをはじめ、各種ウィンドウやアイコンをどこにでも置けるようになっているので、ワークスペースを自由にレイアウト。パーソナルな環境をつくることができます。

NeXT™

Canon

資料請求 No.015

NeXT発。コンピュータの未来をひらく、7つのブレークスルー⑥

テキスト、グラフィックス、さらにサウンドや音声を組み合わせた電子メール。NeXTは、マルチメディアでもあります。

1990年代、NeXT Mailが、コミュニケーションの新しい流れをつくります。

1980年代、デスクトップパブリッシングが画期的なコンピュータ利用法として登場しました。印刷媒体によるコミュニケーションのスタイルを変え、現在では日常的なコンピュータの機能になっています。1990年代、コミュニケーションの領域で、電子メールがデスクトップパブリッシングと同じようなインパクトをもたらすことになるとNeXTは考えました。NeXT Mailは、重要な機能のひとつとしてマシンに組み込まれた電子メールシステムです。そのソフトウェアは、最初からNeXTにバンドルされ、EthernetおよびTCP/IP用のプロトコルが内蔵されているので、拡張カードといったハードウェアを追加する必要もありません。NeXT Mailを、ユーザーは即座に利用することができます。

● UNIXは米国AT&T社の、Ethernetは米国ゼロックス社の登録商標です。

NeXTはマルチメディアへ。サウンドや音声による電子メールも可能です。

NeXT Mailには、NeXTならではの優れた特長があります。まず、NeXTはマルチタスク環境であるため、作動中のプログラムを中断することなく、いつでも電子メールの送受信を行うことができます。また、これまでの電子メールシステムとは違い、さまざまな書体とさまざまなサイズの文字を使ってテキストを送ることが可能。将来の拡張により、メッセージの一部にグラフィックイメージを入れることもできます。デジタル信号プロセッサ(DSP)標準搭載のNeXTでは、さらにサウンドや音声による電子メールも可能です。NeXT Mailは、まさに「しゃべるメール」といえるでしょう。もちろん、従来のUNIXの電子メールとの互換性もあり、テキストによる相互ネットワーク通信ができます。

高度なメール機能を、簡単な操作で。音声入力もテープレコーダ感覚です。

NeXT Mailは、ほとんどマニュアルなしで使うことができます。たとえば、音声入力。「リップサービス」を選択すると、テープレコーダのような操作パネルが現れます。録音ボタンを押し、マイクから音声を入力。録音が終わったら、それをメールメッセージの中に。送信ボタンを押せば、送信完了です。受信者は「リップサービス」のアイコンをクリックするだけでメッセージを聞くことができます。みんなのマルチメディア。それがNeXTです。

NeXT™

Canon

NeXT発。コンピュータの未来をひらく、7つのブレークスルー⑦

高度なソフトウェア開発を
マウス操作で簡単に。
プログラミングのスタイルも、NeXTは1990年代です。

NeXTは、ユーザーだけでなく、プログラマーのニーズにも応えるために。

今日、ソフトウェアの開発は、極めて高度で複雑なものになってしまいました。ソフトウェアを人に使いやすくするためのユーザーインタフェースへの要求が増大したからです。かつてはソフトウェアの開発のほとんどがコア部分で、ユーザーインタフェース部分の開発の割合は非常に小さなものでした。その後、ユーザーインタフェースの比重が高まり、現在では開発時間の大半がユーザーインタフェースのプログラミングに費やされています。しかも、ユーザーインタフェース開発のためには、相当な熟練も必要。操作をやさしくするためのユーザーインタフェースが、プログラミングを難しくしているのです。NeXTは、だれでも使えるソフトウェア開発環境を目指して、NextStepを構築しました。

複雑なユーザーインタフェース開発を、マウス操作のグラフィック方式で。

NextStepは、まったく新しいタイプのソフトウェア開発環境です。ユーザーインタフェースのための、かずかずのサブルーチンに当たるものを、わずか18のオブジェクトからなるアプリケーションキットにまとめました。さらに、ユーザーインタフェース部分のプログラムを設計する場合、インタフェースビルダが利用できます。これは、Objective Cのソースコードを生成する本格的なアプリケーション開発ツールです。オブジェクトとオブジェクトをマウスで結びつけるだけで、ユーザーインタフェースの構築が可能に。また、パレットというインタフェース要素(ウィンドウ、メニュー、ボタンなど)が用意されているので、ユーザーインタフェース開発の大部分を、グラフィック方式で簡単に処理できます。

1990年代。NextStepが、ソフトウェアの世界をどんどん広げていきます。

ユーザーインタフェース開発のためのアプリケーションキットとインタフェースビルダ。そして、ファイル管理を行うワークスペースマネージャーと、すべてのイベントを管理するウィンドウサーバーをくわえたものが、NextStepのアーキテクチャです。NeXTに搭載のNextStepが、プログラミングの時間を大幅に短縮し、より大量の、より質の高いプログラムの獲得を実現。1990年代のソフトウェアの可能性を大きく広げました。

NeXT™

Canon

Sun Application Fair 1990

進む。拡がる。Sun・地球 ネットワークの2日間。

8月30日[木]-31日[金] 新宿NSビル・地下1階 イベントホール〈入場無料〉AM9:30〜PM5:00

sun microsystems

統合化で拓ける。

Sun Sociabilité

統合化が進むことで、人は大きな拡がりを手にすることができます。
Sociabilité――ソシアビリテ。フランス語で社交性、社会性などの意味をもった言葉。
そして、私たち日本サン・マイクロシステムズの、今回のキーワードです。
Sun Sociabilité、それは、オープンシステムが実現する親密なつながり。
人とコンピュータ、コンピュータとコンピュータ、あるいは、コンピュータを介しての、人と人との新しい関係。統合化の波は、地球規模のコミュニケーションに、素晴らしい変化をもたらそうとしています。私たちが、今回のフェアで提案するのは、未来を見据えた、豊かなコミュニケーションの形。Sun Sociabilitéは、すでに多くの実りとともにあります。

■セミナー同時開催

[日　時] 8月30日[木]・31日[金] 10:00〜16:30
[会　場] 東京ヒルトンインターナショナルホテル　4F菊華の間
[内　容]
- ●1990年代のワークステーション　(30日・31日 10:00〜12:00)
- ●ディーリングシステムにおけるワークステーションの利用　(30日のみ 13:00〜14:30)
- ●SPARCの最新動向　(31日のみ 13:00〜14:30)
- ●日本語OpenWindows　(30日・31日 15:00〜16:30)

●主にサン・マイクロシステムズ社より説明いたします。(同時通訳のサービスを行います。)
なお、「ディーリングシステムにおけるワークステーションの利用」はロイター ジャパン㈱様より講演されます。

Sun Sociabilité
進む。拡がる。Sun・地球ネットワークの2日間。
Sun Application Fair 1990

■Sunアプリケーション フェア1990出展企業一覧(50音順)

INSエンジニアリング㈱
㈱アイザック
㈱アスキー
㈱アステック
アストロデザイン㈱
㈱アルメックス
伊藤忠テクノサイエンス㈱
㈱イングレス・ジャパン
インタフェイス コンピュータ ジャパン㈱
エーエスアール㈱
㈱エー エスピー
㈱ABC
㈱エアー
㈱SRA
エス・シー・ハイテクセンター㈱
㈱エヌ・ケー・エクサ
NTTアドバンステクノロジ㈱
㈱エム・ケー・シー
エム・シー・エム・ジャパン㈱
LSIロジック㈱
㈱オーエスアイ・プラス
オートデスク㈱
オーバーシーズ・ベクテル・インコーポレーテッド
オムロン ソフトウェア㈱
㈱オラクル
兼松エレクトロニクス㈱
兼松セミコンダクター㈱
川崎製鉄㈱
㈱計測技術研究所
ケイデンス・デザイン・システムズ㈱
㈱建築システム
㈱神戸製鋼所
国際テレコメット㈱
コニカメディカル㈱
サイバネット システム㈱
㈱ザックス
シーアイ テクノセールス㈱
㈱CSK
シーティーシー・システムズ㈱
シーティーシー・ファイナンシャルエンジニアリング㈱
住商電子システム㈱
㈱システムネットワーク
㈱シナガワ
神鋼電機㈱
新日本製鐵㈱
住商エレクトロニクス㈱
㈱図研
セイコー電子工業㈱
センチュリ リサーチ センタ㈱
ソニー・テクトロニクス㈱
㈱ソフィアシステムズ
㈱ソフトウェア クレイドル
ソフネック㈱
ディジタル サービス㈱
データ コントロール リミテッド
㈱電通国際情報サービス
㈱トータル・マネジメント・サービス
㈱トーメンエレクトロニクス
東京エレクトロン㈱
㈱東芝
東芝エンジニアリング㈱
㈱東洋情報システム
㈱東陽テクニカ
日軽情報システム㈱
㈱ニコン
ニチメンデータシステム㈱
日商エレクトロニクス㈱
日本アプライド・マイクロシステムズ㈱
日本エル・ピー・アイ㈱
日本鋼管㈱
日本サーク㈱
日本システム技術㈱
日本シルバーリスコ㈱
日本タイムシェア㈱
日本バイナリー㈱
日本バリッドロジックシステムズ㈱
日本プライムコンピュータ㈱
日本ユニシス㈱
日本レーカル・リダック㈱
パーソナルメディア㈱
BBNソフトウェア㈱
㈱ビクター・データ・システムズ
ファームウェアシステム㈱
ファモティク㈱
富士ゼロックス㈱
富士通㈱
㈱物産システムテクノロジー
VLSI テクノロジー㈱
㈱マイクロ・シー・エー・デー
丸文㈱
丸紅エレクトロニクス㈱
㈱丸紅ハイテック・コーポレーション
三井造船㈱
三井物産㈱
三菱電機㈱
㈱メイテック
㈱メトロ
ユニソル㈱
㈱LIFEBOAT
ラインシステム㈱
菱洋エレクトロ㈱
㈱リード・レックス
㈱理経
㈱リコー
ロータス㈱
ロイター ジャパン㈱
㈱ワコム

計 107社

日本サン・マイクロシステムズ株式会社

本　　社　〒102 東京都千代田区二番町11-19 興和二番町ビル　電話 (03)221-7021㈹
大阪支社　〒550 大阪市西区京町堀1-4-16 センチュリービル9F　電話 (06)445-5231㈹
名古屋営業所　〒460 名古屋市中区錦3-6-29 興和ビルサウスハウス7F　電話(052)963-8560㈹
つくば営業所　〒305 茨城県つくば市春日3-12-10 高野ビル2F　電話(0298)51-2210㈹

お問い合わせ　日本サン・マイクロシステムズ アプリケーションフェア事務局　電話(03)221-7228

資料請求 No.021

あなたの、そばにエプソン。

これ、頼もしいです。

32ビットの頼もしさ。
ビジネスの最先端で活躍する
僕の高性能な先輩を紹介します。

BUSINESS SYSTEM-1

5 MOD-600SC　4 HDD-100SC　3 SC-1000　1 PC-386V　2 LP-7000

デスクトップパブリッシング

本格DTPだってお手のもの。
SCSI環境フル活用の強力システム。

ますます多様化するビジネスのニーズに応える
32ビット・ハイパフォーマンス・システム。
SCSI環境を生かした大規模&超高速な情報処理が得意技。
印刷並みの美しい文書が作成できます。

1 32ビットCPU386™20MHzノーウェイトの超高速処理を実現。ユーザーズメモリも1.6MB標準実装、EMS機能をサポートしています。2 速く、美しく、静か。B4からハガキまで幅広い用紙に対応。アウトラインフォントを標準装備。解像度240dpiの高品位印字を実現。3 PC-9801-55ボード互換。最大7台までのSCSI周辺機器を接続可能。※ 4 画像処理、データベースなど大量のデータ処理に応える100MBの大容量。平均25msの高速アクセス。5 レーザー光で読み書き可能な光磁気ディスク。SCSI環境を強力にサポートする600MBの大容量。

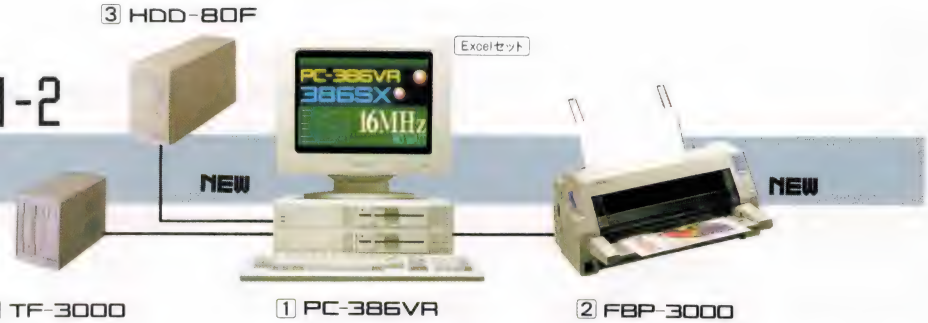
標準価格(税別) 1)パーソナルコンピュータPC-386V-STD ¥498,000 H20 ¥623,000 H40 ¥683,000 2)日本語ページプリンタLP-7000 ¥571,000 3)SCSIボードSC-1000 ¥35,000 4)SCSI対応100MBハードディスクHDD-100SC ¥198,000 5)SCSI対応600MB光磁気ディスクMOD-600SC ¥450,000

※OSにより異なります。

LP-7000には別売のインターフェイスケーブルが必要です

EPSON

BUSINESS SYSTEM-2



伝票処理なら、ドーンとおまかせ。膨大なデータも高速で処理します。

水平ローディング方式のプリンタとの組み合わせて伝票など
様々な用紙に対応。デスクワークをバリバリこなします。
増設FDDでPCノートなど3.5インチパソコンとデータを共有できます。

①CPU386SX™16MHzノーウェイトで使用。豊富な98ソフトはもちろん、MS-WINDOWS対応ソフトなど大きなプログラム、膨大なデータ処理に高速で応えます。②水平ローディング方式の給紙により複写紙など多様な用紙に対応。オリジナル+5枚のコピー能力。7色のカラー印字も可能。③80MBの大容量を平均アクセスタイム25msで高速ドライブ。④3.5インチパソコンとデータを共有することが可能です。

標準価格(税別) ①パーソナルコンピュータPC-386VR-STD ¥348,000/H40 ¥488,000 ②24ピンドットマトリクス漢字カラープリンタFBP-3000 ¥248,000 ③ハードディスクユニットHDD-80F ¥268,000 ④3.5インチフロッピーディスクユニット TF-3000 ¥89,800

プリンタの価格はPCセット(本体+PCシリーズ用インターフェイスケーブル)のものです

BUSINESS SYSTEM-3

書類をつくる、書類を送る。仕事のベテラン、はりきりシステム。

32ビット・ラップトップPCにFAXボードをプラス。
すると、資料や見積書など作った書類をそのままFAXで
送れてしまう。ビジネスをスピードアップするシステムです。

①CPU386SX™16MHzノーウェイトの高速処理。着脱容易な20MB 40MBハードディスクバックを装着可能です。②漢字220字 秒(高速モード)の素早さで24×24ドットの高品質印字。耳さわりな作動音も気にならない静粛設計です。フルオートペーパーハンドリング機能搭載。③パソコンとFAXの間でダイレクトに文書や絵を送れるFAXアダプタ。④MIMアクティブマトリクス方式カラー液晶搭載。アナログRGBに対応した4096色中16色表示。32ビットCPU386SX™16MHzノーウェイトの高速処理。

標準価格(税別) ①パーソナルコンピュータ PC-386LS-STD ¥538,000/H20 ¥673,000/H40 ¥763,000 ②24ドットインクジェット漢字プリンタHG-3000 ¥248,000 ③カードファックス FR-2000 ¥49,800 ④パーソナルコンピュータ PC-386LSC ¥980,000

プリンタの価格はPCセット(本体+PCシリーズ用インターフェイスケーブル)のものです

日本人のためのソフトウェアを追究する。

ASP

日本語スクリーン・エディタ

X-Final

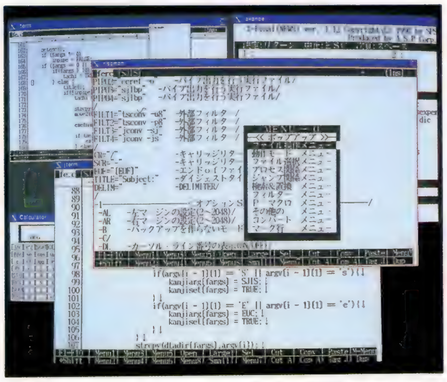
マルチプラットフォームで幅広く利用されているX-FinalのSunview版を発売開始しました。

X-Final Sunview版は、好評のX-Window版同様、UNIXワークステーションに対応した最新の開発環境を提供します。また、漢字コードの自動判別をはじめ、きめ細かい日本語環境をサポートしています。

異なる機種、異なるウィンドウ・システムをつないだネットワーク上で、X-Finalのダイナミックな編集環境をご利用ください。

SunView バージョン

価格¥98,000（消費税別）



新・発・売！

SunView対応機種	X-Window対応機種
● SPARC LT	● NEWS
● AS4000(SPARC)シリーズ	● RISC NEWS
● SPARC-STATIONシリーズ	● LUNA(UNIOS-B)
● AS3000 SUN3	● LUNA SX9100(UNIOS-U)
● SUN SPARC Station版	● Laptop NEWS新発売
● AS4000シリーズ発売中（SPARC）	● EWS4800
● その他	● PC-UX V
	● 2050G
	● NeXT（開発中）

■サイトライセンス契約のご案内■

弊社では、ソフトウエアをより広くご利用いただくために、学校および企業に対しする一括契約を行なっています。現在でも東京大学計算機センター、同志社大学、岡山理科大学、山梨大学、㈱ハドソン、日本電装㈱……などの教育機関や各種の企業でFinalをはじめとするソフトウエアを一括利用していただいています。
どうぞお気軽にお問い合わせください。

※Xウィンドウ・システムは、米国マサチューセッツ工科大学の登録商標です。
UNIXは、米国AT&Tベル研究所の登録商標です。
SUN Viewは米国サンマイクロシステムズ社の商標です

株式会社 エー・エス・ピー
〒143 東京都大田区大森北5-8-11-106
TEL.03-767-1451 FAX.03-767-1453

資料請求 No.024

98用OS/2版
好評発売中!

RED2
The Multi Screen Editor

衝撃の赤いエディタ RED2

SPECIFICATIONS.
MaxFiles:20
MaxNumber of Lines:65535
MaxLength of Line:180
MaxCapacity:5Mbyte
TopScrollSpeed:12000/min
(20MHz386PC)

MS-DOS版　OS/2版
統一価格
¥38,000

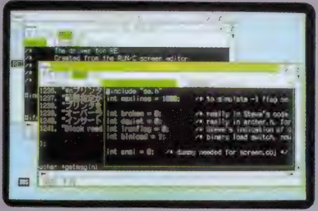
■NEC PC-9801シリーズ(ハイレゾ/ノーマル共用、LTを除く)
富士通 FMシリーズ(中解像度/高解像度共用)
松下 Panacom Mシリーズ(中解像度/高解像度共用)
■MS-DOS版 384K以上
OS/2版 4M以上
■MS-DOS版 各機種共通 MS-DOS Ver2.11以上
OS/2版 FMシリーズ&Panacom Mシリーズ MS OS/2 Ver1.0以上
PC-9801シリーズ MS OS/2 Ver1.1以上
(ご注文の際には機種名とOSの別をご指定ください)

新世代プログラミング環境を提供するRED2(レッド・ツー)。最先端エディタに要求される絶対条件の限りない追及によって、あなたの意志に忠実に応える操作環境を実現。衝撃の赤いエディタREDとして1985年に発表されて以来、その蓄積された技術の集大成として、極限まで最適化されたレスポンス性/操作性。ビルト・インされた新機能の数々。より強力なユーザ・インターフェースの実現。様々なシーンで確実に追従する操作性は、キーボードに触れるあなたにRED2との一体感、調和感を与えます。

■マルチウィンドウ/マルチファイルエディタ
RED2は、最高20ファイルを同時オープン。各ファイルを独立したウィンドウに割り当てます。各ウィンドウは、表示位置やサイズを自由にレイアウト可能。複数のファイルを参照しながらテキスト入力が行えます。また、編集ファイルをコピーした複製ウィンドウも作成可能。テンポラリ・ウィンドウとして様々な用途に活用できます。

■ハイレゾモード/ノーマルモード自動判別機能搭載
PC-9801版では、ハイレゾ/ノーマルに対応した31行モード/25行モードを自動切り替え可能。FM版では、高解像度/中解像度を自動判断し、正しいテキストRAM位置を認識。高性能デスクトップマシンからラップトップパソコンまでフル対応します。(自動判別機能はMS-DOS版のみサポート)

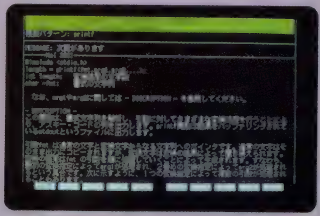
■プレゼンテーション・マネージャに対応するOS/2版
RED2 OS/2版はMS-DOS版と上位互換。MS OS/2 Version1.1に搭載されたプレゼンテーション・マネージャにも、フルスクリーンモードにも対応。複数のウィンドウで同時編集を行ったり、他のセッションと同時に起動することが可能です。



■内蔵タグジャンプ機能
C言語プログラミングに便利な内蔵タグジャンプ機能搭載。編集中のC言語ソースファイル内に記述された様々な関数の一覧を表示します。表示された関数一覧の中から任意の関数を選択することにより、クイックに目的の関数までカーソルを移動させることが可能です。(PC-9801シリーズのみサポート)

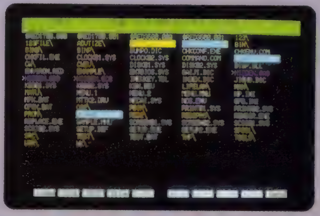
■[illegible]
別ファイル内で定義されたC関数まで、カーソルを移動するダイレクトタグジャンプ機能に加え、ジャンプした元の位置まで戻るバックタグジャンプ機能を追加。複数のソースファイルを縦横無尽に飛び交うことができます。

■C関数の仕様を表示するガイダンス機能
C言語プログラミングを大幅に効率アップするガイダンス機能を搭載。プログラミング中にいつでもC言語関数リファレンスを表示します。ANSI-Cに準拠した400種以上の関数が登録済み。ユーザによるリファレンス情報のリメイクが可能。他言語への対応や、既存リファレンスのカスタマイズも容易です。



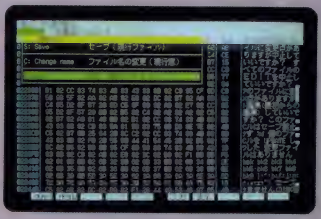
■編集ファイルのバージョン管理機能
最高9世代のバックアップファイルを作成するファイル管理機能搭載。オペレーションミスによるファイル損失への対応、編集&セーブを繰り返すプログラム開発の保守に威力を発揮。また、現在編集中のファイルを最初から編集し直すオリジナルファイルのオープン機能も搭載しています。

■ディレクトリ/ドライブの移動が可能なファイル選択機能
ハードディスク時代にマッチするツリー構造対応型のファイル選択機能搭載。広大なディスク空間に分散された編集ファイルをディレクトリを上下に移動しながらオープンします。また、リードオンリーファイルや、ヒドゥンファイルの確認も可能。ディレクトリ一覧表示によるクイックなファイルオープンができます。



■さまざまなマクロ機能を搭載
C言語の『main(argc,argv)』など、定型が決った文字列を任意のキーに割り当てるパーマネントマクロ機能、キーオペレーションのヒストリーを記憶するログ機能など、キータッチを軽減するマクロ機能を搭載。また、ログ機能によって作成された情報をファイルにセーブするログセーブ機能を追加。いつでも以前のオペレーションを再現可能です。さらに、RED2起動直後にログ情報に基づいたオペレーションを行い、変更ファイルをセーブしたのちに、DOSに戻る自動実行機能も搭載しました。

■[illegible]
COMファイルやEXEファイルの編集を行うバイナリエディットをサポート。16進数で表示されるダンプリストと、それに対応するテキストを同時表示する2モード方式を採用。16進数入力も文字入力も可能。また、全角文字表示とグラフィック文字表示の切り替えも行えます。



■C言語/アセンブラ編集時に便利なプログラマーズ電卓
アセンブラやC言語のプログラミング時など、計算を行う場合に便利なプログラマーズ電卓搭載。2進、8進、10進、16進数をダイレクトに変換することが可能。計算途中の値を一時的に保管するメモ機能も搭載。計算結果をテキストに直接転送することも可能です。

■[illegible]
RED2は、文字列の検索/置換時に区切り文字を識別。例えば『getc』を置換する場合、『getchar』を検索対象から外すことができます。また、正規表現をサポートするGREPを活用することにより、さらに高度な検索も可能です。

■[illegible]
BASICやCOBOLなどに利用可能な行番号生成機能を搭載。行番号をオートマチックに各行に付属させることが可能です。

■LIM EMS 4.0対応
RED2が作成するテンポラリファイルの内容をEMSメモリ上に割り当て可能。ディスクアクセスを軽減した高速処理を実現します。(PC-9801シリーズMS-DOS版のみサポート。推奨EMSボード&ドライバ:I・O DATA PIO-PC34シリーズ/PIO-RA34-3M/PIO-RL34-4M&IOS-10EMS/386又はIOS-10Ⅱ)

■様々な日本語フロント・エンド・プロセッサに対応
RED2は様々な日本語フロント・エンド・プロセッサに対応。プログラミングのみならず、高速な簡易ワープロとして使用することもできます。付属ユーティリティLPRの使用によるアンダーラインや横倍角文字を使用したプリンタ出力、カットバッファ内のテキストのダイレクトプリンタ出力も可能。文書作成を強力にサポートします。

●横スクロール180桁サポート(80桁~180桁を任意指定可能)●超高速スクロールモード●子プロセス実行時のヒストリー機能を強化●直前に変更した位置にジャンプするラストタッチ機能●矩形カット&ペーストを強化●バックスラッシュ表示モード●ログの連続実行●文字列範囲指定で指定文字まで自動移動●カットバッファ内容の表示●カットバッファのプリンタ出力●テキスト中の文字コード表示●漢字/ACCIIコード表示●ガイダンスファイルの自動リメイク機能●拡張命令モードキー設定●自動コンパイル機能●テキストをDOSのコマンドとして実行etc...

【お求め方法】 当社製品は、パソコンソフト取扱い店でお求めいただけますが、弊社に直接ご注文の際は、The Programmer's INDEX (TEL:03-293-3887)までお問い合わせ下さい。 ※各製品の表示価格には、消費税は含まれておりません。

●Copyright Notice
＊MS-DOSは米国マイクロソフト社の登録商標です。
＊RED2, RED++は株式会社LIFEBOATの商標です。
＊MS®OS/2は米国マイクロソフト社の登録商標です。

株式会社 LIFEBOAT
〒101 東京都千代田区神田錦町3-6
PHONE:03-293-4711(代) FAX:03-293-4710

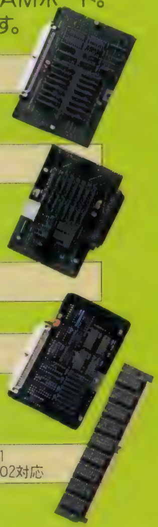
I·O DATA

32ビット機専用だから、この高速、この安心。

思い通りのフレーズがすらすらと弾けてしまう時って、
ついギターをよしよしってしたくなりますね。
自分のイメージ通り、素速く、的確に応えてくれるアイ・オー。
ついつい自慢したくなっちゃいますね。

超高速で強力サポート。
32ビットマシンの最大限のパワーを引き出します。

32ビットパソコンの活用は、メモリ選びがポイント。アイ・オーの32ビット専用RAMボードなら、高度な仮想EMSを超高速で実現します。例えばお手持ちの「一太郎Ver4」「花子」「ロータス1·2·3」など、EMS対応アプリケーションソフトを超高速でご利用いただけます。386CPUによってはじめて可能になる高機能環境を32ビットパワーでフルサポートするアイ・オーのRAMボード。専用だから、32ビットパソコンの"高速性"を最大限に引き出します。



RA21/51専用増設RAMボード PIO-RA134 PC-9801RA21/51対応

2Mバイト(PIO-RA134-2M)……¥38,000　6Mバイト(PIO-RA134-6M)…………¥115,000
4Mバイト(PIO-RA134-4M)……¥77,000

RS21/5専用増設RAMボード PIO-RS134 PC-9801RS21/51対応

2Mバイト(PIO-RS134-2M)……¥38,000

■PIO-RA134/RS134用増設オプションPIO-EX134
2Mバイト(PIO-EX134-2M) ¥40,000　4Mバイト(PIO-EX134-4M) ¥80,000

RL2/5専用増設RAMボード PIO-RL34 PC-98RL2/5対応

4Mバイト(PIO-RL34-4MA)……¥74,000

ES2/5専用増設RAMボード PIO-ES34 PC-9801ES2/5対応

1Mバイト(PIO-ES34-1MA)……¥26,000　3Mバイト(PIO-ES34-3MA)…………¥60,000

■PIO-ES34-1MA用増設オプション PIO-EX34SX-2MA 2バイト¥39,000

2MB増設サブボード PIO-SIM61 PC-9801RA-11/RS-11/T-01 PC-H98/70-B01、PC-H98-B02対応

2Mバイト(PIO-SIM61-2M)……¥42,000

クロード・チアリ
音楽家としてだけでなく、パソコンをなくして生活も仕事もできないほどのパソコン好きで有名なクロード・チアリさん。パソコン通信で日本一のメンバーを持つ「NEC98 by チアリ」(PC-VAN)を主宰。

パソコン増設RAMのトップブランド
株式会社アイ・オー・データ機器

本　　社／〒920 石川県金沢市駅西本町1-5-41
TEL(0762)21-4812　FAX(0762)24-9300
東京営業所／〒101 東京都千代田区神田富山町6松崎ビル4F
TEL(03) 254-0301　FAX(03) 254-9609
大阪営業所／〒532 大阪市淀川区西中島7-14-35大阪屋北5号館
TEL(06) 303-3010　FAX(06) 303-3110
※広告掲載の表示価格には消費税は含まれておりません。

新製品 ハード・ソフト 激安 販売中!! 是非、御来店下さい!

PRO SHOP ツクモ5号店

SUPER ASCII 創刊記念第2弾!!

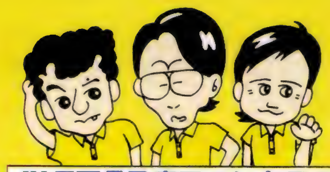
グラフテックフェアー第3弾開催中! MO・光ディスクツクモ5号店

貴方のお持ちのコンピューター高く買い取ります!

全国代金引き換え配達も行います!

学校・会社関係に卸し販売致します!

各種カード・クレジット・リース取り扱いOK!



ツクモ5号店ファンクラブ ☎03(251)0531

マイコンは買ってしまえば終り?そんなはずはありませんよね。ツクモ5号店では所有されてからの貴方を応援します。消耗品、ソフトを会員ならではの特別プライスで提供いたします。★お買物の際、アンケートにお答え頂くだけで即ご入会できます。入会お待ちしております。

新/製/品/情/報

人気No.1 ツクモオリジナルドライブ

2HD専用タイプ

2DD/2HD自動切替タイプ

2HD専用 TS-SRタイプ	定価	特価
TS-5SR1	¥42,800	¥35,800
TS-5SR2	¥63,800	¥53,800
TS-3SR1	¥36,800	¥29,800
TS-3SR2	¥56,800	¥45,800

自動切替 TS-ERタイプ	定価	特価
TS-5ER1	¥51,800	¥42,800
TS-5ER2	¥72,800	¥60,800
TS-3ER1	¥44,800	¥37,800
TS-3ER2	¥64,800	¥54,800

2HD専用 TS-STタイプ	定価	特価
TS-5ST(F)	¥48,800	¥29,800
TS-5ST(2)	¥78,800	¥46,800
TS-3ST(F)	¥36,800	¥25,800
TS-3ST(2)	¥68,800	¥41,800

自動切替 TS-EXタイプ	定価	特価
TS-5EX(F)	¥62,800	¥37,800
TS-5EX(2)	¥92,800	¥55,800
TS-3EX(F)	¥49,800	¥33,800
TS-3EX(2)	¥82,800	¥49,800

数値演算データプロセッサーツクモ特価販売中!!

i8087-1 (PC-9801/E/F/M/VF/VM、PC-286/U/US用) 定価¥39,000 特価¥27,800

i80287-10 (PC-286/V/VE/VF/UX、PC-9801/VX/RX/LX用) 定価¥59,000 特価¥33,800

i80387-16 (PC-9801RA用) 定価¥90,000 特価¥54,800

i80387SX-16 (PC-9801/ES/RS/LS、PC-386M/VR用) 定価¥110,000 特価¥59,800

i80387-20 (PC-9801RA21、PC-386V用) 定価¥110,000 特価¥69,800

i80387-33 (HYPER98用) ツクモ特価販売中!!

★INTEGRATEG INFORMATION製の2C87、3C87もあります。

フレームバッファ大特価

ハイパーフレーム(+) 定価¥98,000 ツクモ特価¥80,500

ハイパービジョン(+) 定価¥98,000 ツクモ特価¥80,500

HYPER-彩子 定価¥88,000 ツクモ特価¥72,500

麗　子 定価¥88,000 ツクモ特価¥72,500

大特価40MBハードディスク

ウインテク WD-40 ……定価¥102,000 特価¥59,800

緑電子 Little E40mkⅡ ……定価¥98,000 特価¥64,800

大容量SCSIハードディスク大特価販売中

メーカー	製品	定価	特価
アイテック	IT-H100G	¥238,000	¥145,000
日本テクサ	STATION80S	¥148,000	¥115,000
	STATION120S	¥178,000	¥140,000
	(I/F HA-55S)	¥25,000	¥19,800
キャラベル	AV-090SC(I/F込)	¥188,000	¥148,000
	AV-130SC(I/F込)	¥228,000	¥180,000
ICM	SRC-80/81	¥158,000	¥125,000
	SRC-131	¥198,000	¥159,800
	(I/F IF-2750)	¥30,000	¥25,000
テクノジャパン	PD-100RS	¥168,000	¥133,000
アークウッド	AW-N100SC	¥195,000	¥145,000
ニューテック	NT-100-55	¥198,000	¥155,000
ニューテック	NT-100-55N	¥208,000	¥165,000

モデム大特価販売中

一流メーカークラス4 ツクモ特価¥28,000

NSE 仲人No.1 定価¥16,800
デジタルホン以外のビジネスホンでOK!(抜群の携帯性)
ツクモ特価¥14,000

NEC 98NOTE & ドライブセット

Aセット

- PC-9801N …… ¥248,000
- TS-5ST(F)NT …… ¥51,800
- RCN-2000 …… ¥39,800
- MD-2HDPLUS(10枚)サービス

合計定価¥339,600

ツクモ特価¥260,000

クレジット例(30回払・税込)
初回¥13,065+月々¥10,400×29回

Bセット

- PC-9801NS …… ¥298,000
- TS-5ST(F)NT …… ¥51,800
- RCN-2000 …… ¥39,800
- 一太郎dash …… ¥39,000
- MD-2HDPLUS(10枚)サービス

合計定価¥428,600

ツクモ特価¥350,000

クレジット例(36回払・税込)
初回¥14,297+月々¥11,900×35回

東芝 DynaBook用

特長 2HD/2DD自動切替。1D、2Dの読み込み可能。

TS-5J3100(F) 定価¥49,800 ツクモ特価¥31,800

TS-3J3100(1T) 定価¥32,800 ツクモ特価¥25,800

SONY Quarter L用

特長 1フロッピータイプ用。2HD用(1D、2D読み込み可)

TS-5QT(F) 定価¥49,800 ツクモ特価¥42,800

増設RAMボード

(98NOTE用EMSボード)

メルコ

RCN-1000 (1MB) ……定価¥26,000 特価¥22,000

RCN-2000 (2MB) ……定価¥39,800 特価¥33,800

(ダイナブック用RAMカード)

メルコ

RCJ-2000 (2MB) ……定価¥49,800 特価¥39,800

IOデータ

PIO-31SS-2MA (2MB) …定価¥46,000 特価¥39,800

(EMSラムボード)

メルコ

EMJ-2000mkⅢ (2MB) ·定価¥44,800 特価¥32,800

EMJ-4000mkⅢ (4MB) ·定価¥79,800 特価¥59,800

冬のボーナス一括払受付中!詳しくはお問い合せ下さい。

ビジネスソフト決算特価大放出! 本数限定

ワープロソフト

- 新松 ……ツクモ5特価¥37,500
- n松 ……ツクモ5特価¥24,500
- AS. NOTE ……ツクモ5特価¥13,800
- P1. EXE ……ツクモ5特価¥29,800
- P1. EXE Plus ……ツクモ5特価¥34,800
- P1. EXE NOTE ……ツクモ5特価¥24,800
- VJE-PEN V2.1 ……ツクモ5特価¥25,600
- VJE-βV.2.5 ……ツクモ5特価¥12,500
- アシストシリーズ全品 ……ツクモ5特価¥7,500

毛筆ソフト

- 毛筆ワープロJR ……ツクモ5特価¥24,800
- 毛筆ワープロSR ……ツクモ5特価¥55,800
- 遊名人Ver3 ……ツクモ5特価¥21,800

エディタ

- VZエディタV1.5 ……ツクモ5特価¥7,900
- MIFES98 Ver5.1 ……ツクモ5特価¥28,800
- RED2 ……ツクモ5特価¥29,800
- 章子の書斎 ……ツクモ5特価¥19,800

レイアウト・文字加工ソフト

- Lotus Freelance ……ツクモ5特価¥58,000
- Z's word JG Ver2 ……ツクモ5特価¥39,800
- プリントショップII ……ツクモ5特価¥14,800
- HYPER PRINT2 ……ツクモ5特価¥39,800
- 言図絵巻 ……ツクモ5特価¥29,200
- プリント専科II ……ツクモ5特価¥9,500

アイディアプロセッサ

- IDOQ2 ……ツクモ5特価¥20,800
- 直子の代筆Ver3 ……ツクモ5特価¥7,500

グラフィックソフト

- Z's STAFF Kid98 ……ツクモ5特価¥19,700
- Z's デジタルクラフト ……ツクモ5特価¥25,000
- CANDY4 ……ツクモ5特価¥40,600
- Thirdy ……ツクモ5特価¥28,000

データベースソフト

リレーショナル

- d:BANEIIIPLUS ……ツクモ5特価¥163,800
- dBXL ……ツクモ5特価¥42,800
- 桐Ver2 ……ツクモ5特価¥63,800

カード型

- THE CARD3 ……ツクモ5特価¥32,500
- CARD3事例集 ……ツクモ5特価各¥6,700
- CARD3ポーター ……ツクモ5特価¥7,400
- CARD3プリント ……ツクモ5特価¥14,400
- NINJA3PRO ……ツクモ5特価¥21,800
- Let'sアイリスV3.0 ……ツクモ5特価¥25,800

その他

- 知子の情報Ver2 ……ツクモ5特価¥20,600
- Technote ……ツクモ5特価¥23,400
- MET'Sname&address ツクモ5特価¥7,800

表計算グラフ

- Lotus1-2-3 R2.2J ……ツクモ5特価¥58,000
- RECALC ……ツクモ5特価¥40,800
- MS-Excel ……ツクモ5特価¥58,000
- MET'S Excel Form ……ツクモ5特価¥14,400
- MS-マルチプランVer4.1 ツクモ5特価¥42,800
- MS-チャート ……ツクモ5特価¥31,600
- THE GRAPH ……ツクモ5特価¥25,000

通信ソフト

- まいとーく ……ツクモ5特価¥18,100
- CCT-98II ……ツクモ5特価¥11,200
- パーティー ……ツクモ5特価¥16,800
- BTerm ……ツクモ5特価¥12,800
- CTerm2 ……ツクモ5特価¥12,300
- ESTerm2 ……ツクモ5特価¥17,000
- COMNET ……ツクモ5特価¥19,800

★他J-3100シリーズ用、AXマシン用のソフトも常時在庫してあります!

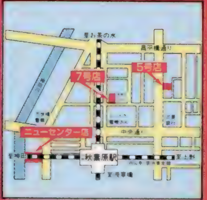
広告掲載のお求めにかんしては、通信販売でもお受けします。記載のない商品に関しては、5号店へお問い合せ下さい。

通信販売部☎03(251)9911

商品のご注文は通販受注センター フリーダイヤル 0120-377-999

お支払方法いろいろ ●現金書留 ●代金引き換え ●銀行振込 ●カード通販 ●クレジット

ツクモ5号店



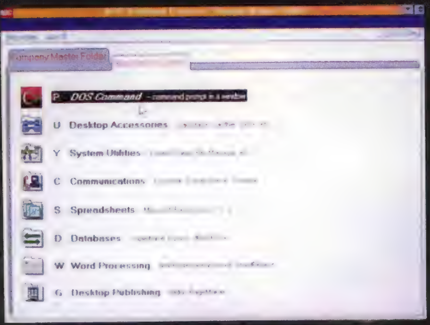
PRO STAFF ツクモ5号店 ☎03-251-0531

ニューセンター店 ☎03-251-0987

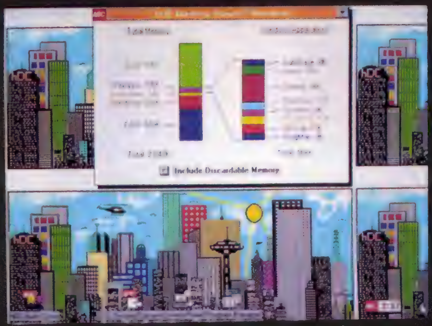
ツクモ7号店 ☎03-253-4199

★表示価格に消費税は含まれておりません。※商品のご注文は在庫確認の上、お願いします。

資料請求 No.027



hDC EXPRESSはWINDOWS 3にメニューシステムを提供。

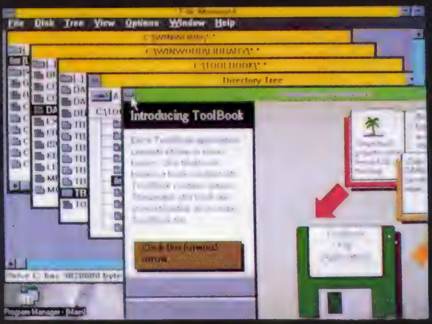


hDCのMemory Viewerでメモリーをグラフィカルに表示。

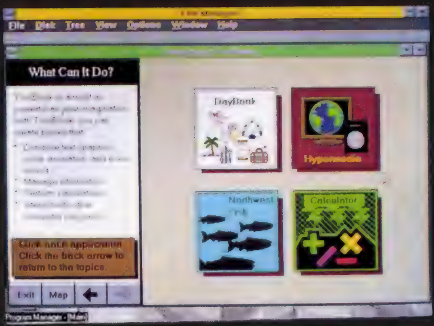
さまざまなプラットフォームで働くWINGZが使用できます。

PS/55ZとWINDOWSが変えるパソコンの世界。

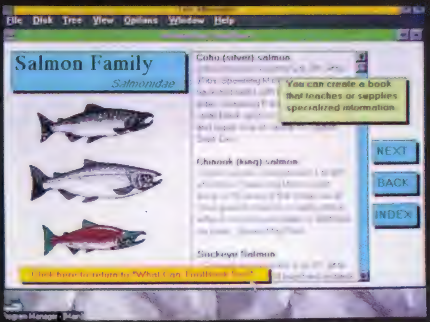
新しいオペレーティング環境、WINDOWS。ディスクオペレーティングシステムの知識なしに
パソコンを操作できるGUI［グラフィカル・ユーザー・インターフェース］。パーソナルでありながらメインフレームのデータ活用も同時にこなすマルチタスク。
OS/2の時代でも共存しながら、未来へと継承されていくWINDOWSの快適な操作環境。日本語、英語を瞬時に使いこなす最新の
32ビットマシン、IBM PS/55ZがこのWINDOWSのすべてをお届けします。



WINDOWS 3は多様なマルチウィンドウ機能を搭載しています。



Tool Bookはマルチメディアにも対応しています。



Tool BookはWINDOWS 3にハイパーカード的な環境を実現。

STAN

IBM PS/55Z
- 32ビットCPU80386SX マイクロチャネルアーキテクチャー
- 主記憶標準2M最大16M
- 1024×786ドット12インチ

高速32ビットパソコンです。

32ビットマイクロプロセッサー80386SXやマイクロチャネルなどのさまざまな先進技術により実現した高速処理。優れた操作性。このIBM PS/55Zを25%引きでお届けします。

ソフトウェアが豊富に揃っています。

日本語モード、英語モードを自由に切り替えて利用できるIBM DOSバージョンJ4.0を採用しています。日本のソフトウェアはもちろん米国のソフトウェアも格安価格で揃えています。

画面ガイダンスで操作が簡単です。

そのままお使い頂けるように画面ガイダンスで操作できるソフトウェアをインストールしてあります。初心者の方もスイッチを入れた途端にパワーユーザーです。

フリーダイヤルで完璧にサポートします。

ご購入いただいたハードウェア、ソフトウェアに対するご質問は、当社カストマーセンターにて常時承っております。フリーダイヤルですのでお気軽にご相談ください。

スタンビジネスシステムはIBM PS/55Zのシステムインテグレーターです。

システム装置

2FDキット［25%OFF］

2FD＋ディスプレイ＋DOS＋キーボード2型

¥476,000 → **¥357,000**

30Mハードディスクキット［25%OFF］

30MHD＋ディスプレイ＋DOS＋キーボード2型

¥626,000 → **¥469,000**

周辺機器

プリンタオート［25%OFF］

5572B02プリンタ＋ケーブル

¥94,800 → **¥71,000**

マウス［25%OFF］

¥15,000 → **¥11,000**

ビジネスソフトウェア

一太郎Ver.4	¥68,000
LOTUS1-2-3	¥98,000
花子Ver.2	¥68,000
CANDY3	¥60,000
Technote	¥48,000
The CARD3	¥48,000

各ソフトを格安価格で揃えております。詳しくはフリーダイヤルでお問い合わせください。

システム装置、周辺機器、ソフトウェアの中から自由に組み合わせてお申し込みください。
あらかじめソフトウェアの導入サービスもいたします。
その他ソフトウェア、ハードウェア、周辺機器など多数ございます。詳しくはカタログをご請求ください。

WINDOWS3ソフトウェア（英語版）

MS-WINDOWS3	¥29,800
TOOLBOOK	
EXCEL	¥99,000
MS-WORD	¥99,000
hDC EXPRESS	¥24,800
hDC First APPS.	¥34,800

各ソフトを格安価格で揃えております。→詳しくはフリーダイヤルでお問い合わせください。

WINDOWS2.11ソフトウェア（日本語版）

WINDOWS2.11	¥20,000
VJE β	¥18,000
hDC EXPRESS	¥24,800
hDC Manager	¥34,800
EXCEL	¥98,000
日本語GUIDE	¥50,000

各ソフトを格安価格で揃えております。→詳しくはフリーダイヤルでお問い合わせください。

日本信販の分割払い12～48回がございます。
JCB、VISAなどのクレジットでのお支払いも承っております

＊IBMは、IBM Corp.（米国）の登録商標です。
＊パーソナルシステムPS/55Zは、IBM Corp.（米国）の商標です。
＊WINGZは米国Informix Software社の登録商標です
＊記載されているソフトウェアの製品名は、各社の商標、または登録商標です。
＊記載されている価格には消費税は含まれておりません。

お申し込み・お問い合わせは
フリーダイヤルにてお受けしています。

0120-225355

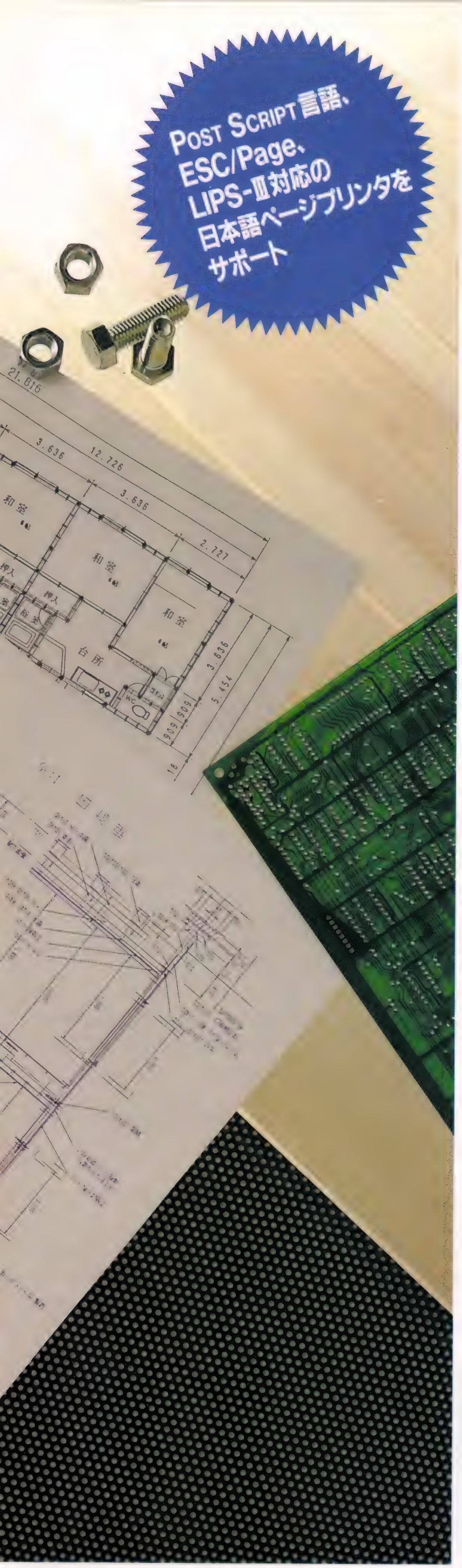
受付時間 月曜～金曜日10:00am～6:00pm

●日本アイ・ビー・エム ビジネスパートナー

スタンビジネスシステム株式会社

160 東京都新宿区新宿5-8-14 phone: 03-225-9485
771-01 徳島市川内町平石流通団地21 phone: 0886-65-5527

※CANDY4には、MS-DOSを組み込んでいません。ご使用になるためには、日本電気株式会社製の日本語MS-DOS Ver.2.0、3.1、3.3のいずれかが必要です。※CANDY2、CANDY3、CANDY4は、株式会社アスキーの製品です。※Thirdy、3・D PERSは、株式会社アスキーの商標です。※Z'S STAFF KiD98、Z'S word JGは、株式会社ツァイトの商標です。※Auto CADは、オートデスク株式会社の商標です。※MS-DOSは、米国マイクロソフト社の登録商標です。※Post Scriptは、米国Adobe Systems社の登録商標です。※ESC/Pageは、セイコーエプソン株式会社の商標です。

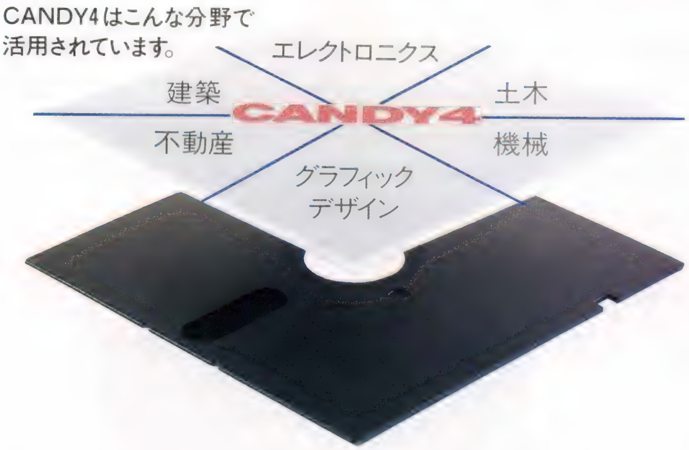


簡単に、早く、正確に。実用本位のドラフティングツール。

製図板を使用するよりも、簡単に、早く、正確に図面が描ける。しかも、必要な図面は保存して再使用できる。本来CADとはドラフティング機能が中心にあるものです。CANDY4は、ドラフティングツールとしての操作性・機能性が高レベルであるとともに、手頃な価格を実現。マウスを使って絵を描く感覚で図面を作成できます。もちろん正確な作図を実行。2way作図機能によりキーボードからの数値入力でも作図できます。さらにはZ'sSTAFF KiD98、Z'sword JG、Thirdy、3・D PERS（近日発売：3次元イメージングプロセッサ）でデータを利用できるオープン環境。DXFファイル出力のサポートにより、AutoCADでもデータを活用できます。建築、土木、機械、エレクトロニクスなど、多くの分野で支持を受け、シリーズ通算10万本の販売記録を達成したCANDYシリーズの頂点、CANDY4。あなたの探していた強力なドラフティングツールがここにあります。

これがCANDY4のアピールポイントです。

●図面データの関連づけができるハイパーパック機能●指定した作図エリアに一気にジャンプできるマーク設定●図形どうしを確実に接続する端点接続●思い通りに移動・回転できる基準点移動●正確な作図や移動ができる詳細表示●メッシュを作図内容に合わせてずらせるメッシュ移動●図形の変形も思いのままにできるリアルタイムサイズ表示●レイヤー間でバック図形も自由に移動できるレイヤー移動●実寸をベースにした正確な作図機能●280機種以上のプロッタ/プリンタをサポート。そのすべての機種で正確な縮尺でプリントアウト●最大8階層のレイヤー機能●図形を自由に拡大縮小できる簡易パラメトリック処理●文字の拡大、回転、変形が自由にできるベクトルフォント●作成した図形を登録、繰り返して使えるパーツケース機能



アスキーの2次元パーソナルCADソフト［キャンディ4］

CANDY4

価格：58,000円（送料：1,000円）表示価格に消費税は含まれておりません。
対応機種：NEC PC-9801VM/UV/VX/UX/CV/LV/LS/RA/RX/LX/EX/ES
PC-98XL²/RL（ノーマルモード）
メディア：5-2HD、3.5-2HD ※マウス、高解像度カラーディスプレイが別途必要です。

CANDY2、CANDY3をお持ちの皆様へ

CANDY4へのバージョンアップのご案内をお送りいたします。まだユーザー登録がお済みでない方は早急にユーザー登録カードをご返送ください。

バージョンアップ料金（詳細は下記営業本部まで電話でお問い合わせください）

CANDY3→CANDY4 10,000円（税込） CANDY2→CANDY4 25,000円（税込）

〒107-24 東京都港区南青山6-11-1 スリーエフ南青山ビル **株式会社アスキー**
お問い合わせ先：営業本部 東京(03)486-8080 大阪(06)348-0018

資料請求 No.031

最新情報を見のがしていませんか?

PARKER BROTHERS & ASCII PROJECT

海外セミナーレポートサービス(限定販売)

㈱アスキーとデータフロー・インターナショナル㈱が共同で、海外セミナーのレポートサービスを致します。欧米にて開催される年間40以上の会議、セミナーから6セミナーを選出し、最小のコストで会議参加と同様の詳細なレポートをお届けします。

システム

- 毎月1回1レポート
- 入会金5万円で第1回レポートを、第2回より毎月8万8千円で1レポートをお届けします。
- 中途申し込み不可

入会金	5万円(第1回レポート価格)
価格	1レポート8万8千円 (第2回レポート以降)
販売形態	会員制による限定販売
製作販売	A4判、約700ページ
申し込み	締切り日 '90 9/10

申し込みとお問い合わせ(企画内容物の)は…

月曜日~金曜日　9:30~17:30
PARKER BROTHERS & ASCII PROJECT
㈱アスキー内　担当　斉藤
港区南青山6-12-1 大仁堂ビル　03-406-8824

入会後の支払い等に関するお問い合わせは…

月曜日~金曜日　9:30~13:30
データフロー・インターナショナル㈱　担当　星野
横浜市栄区上郷町2173-13　045-894-5466

- 上記価格に消費税は含まれておりません。
- 入会金の返却は致しませんが、第1回レポートはお送り致しますのでご了承下さい。
- お支払い方法等につきましては、入会お申し込み後、別途ご案内を致します。
- 会員数により当企画中止の場合がありますのでご了承下さい。

レポートスケジュール

8月　第1回　The Fifth International CD-ROM Conference & Exposition　サンフランシスコ'90/2月
セミナー参加企業情報を含める詳細な情報が満載の約1,500ページのレポートです。(写真)

9月　第2回　Multimedia Expo　ニューヨーク'90/3月
Technology for the 1990's MULTIMEDIA
その動向は世界中が注目しました。

10月　第3回　SIGGRAPH　ダラス'90/8月
Association for Computing Machinery主催の本会議レポート。コンピュータグラフィックス開発における最新情報をお届けします。ビデオレポート予定。

11月　第4回　EuroInfo　ローマ'90/9月
東ヨーロッパ市場の現状と、今後のビジネスチャンスに関する最新情報をお届けします。

12月　第5回　CD-ROM Expo　ボストン'90/10月
今年の会議はこれまでの最大規模のものといわれています。

('91) 1月　第6回　Intertainment　ニューヨーク'90/10月
インタラクティブマルチメディアは、娯楽センターから情報通信企業まで、幅広い業種で利用できるメディアです。本会議では、新しく開発され将来の市場を形勢するようなものについて、特にスポットを当て取り上げます。ビデオレポート予定。

郵便はがき

料金受取人払

赤坂局承認

2257

差出有効期間
平成４年６月
30日まで
（切手不要）

107-24

東京都港区
南青山6-11-1
スリーエフ南青山ビル

株式会社アスキー
広告部
「資料請求」係行

あなたの欲しい資料はすべて入手可能です.

- 資料請求カードは，掲載広告に関する資料集めです．どうぞお気軽にご利用ください．
- なお，ご希望の資料がお手もとに届くまで，１ケ月ほどかかります．あらかじめご了承ください．
- カードの記入方法は，208ページをご参照ください．

郵便はがき

料金受取人払

赤坂局承認

2264

差出有効期間
平成３年12月
31日まで
（切手不要）

107-24

（受取人）

株式会社アスキー
スーパーアスキー
編集部行

SEP

月刊スーパーアスキー編集部では，読者の皆様の声をお聞きして，内容をさらに充実したものにしたいと考えています．表面の空白部分には，お便りやイラスト，スーパーアスキーへのご意見・ご感想などを自由にお書きください．

1990年
9月号

ASCII アンケートハガキ

●HBなどの濃い鉛筆でご記入ください。

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	15	16	17	18	19	20	21	22	23	24	25	26		
2	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	15	16	17	18	19	20	21	22	23	24	25	26		
3	1	2	3	4	5	6	7	8	9	10	11	12	13	
4	1	2	3		5	1	2	3	4		6	1	2	3
7	1	2	3	4	5	6	7	8	9	10	11	12	13	
8	1	2	3	4	5	6	7	8	9	10	11	12	13	14
9	1	2	3	4	5	6	7	8	9	10		10	1	2
11	1	2	3	4	5	6	7	8	9	10	11	12	13	

フリガナ お名前	
ご自宅の住所	〒　都道府県　市区郡 ☎(　)
使用機種	自宅 メーカー 機種 勤務先 メーカー 機種
読者プレゼント希望No.(商品名)	

アンケート記入方法は191ページをご覧ください。

毎月10日までに到着したハガキの中より、抽選で50名の方に図書券500円分をさしあげます（ただし抽選は「今月のプレゼント」を優先させます）。

資料請求カード ASCII 9月号 (AI) 1990年11月15日まで有効

ご面倒でも、フリガナ・送付先ご住所等、全項目にご記入ください。

※お問合せ先: TEL.03-797-3464広告部まで

フリガナ 姓名		生年月日19　年　月　日	性別 男・女　年齢　歳
送付先ご住所 1.自宅 2.勤務先	〒　TEL.(　) 都道府県　区市郡　区町村		
勤務先名／学校名		所属・役職／学部	
職種	1.コンピュータ関連技術者　2.1以外の技術者　3.営業職　4.事務職　5.管理職　6.公務員　7.学生　8.その他		
使用機種	自宅 メーカー 機種名 勤務先 メーカー 機種名		

1.2.で囲んでください。

No.	No.
No.	No.
No.	No.
No.	No.
No.	No.

カードの記入方法は、208ページをご参照ください。

Super EXPRESS

SEPTEMBER 1990

34 日本電気が低価格の日本語PostScriptプリンタなどを発売

35 AJがMacintosh用の低価格レーザープリンタを発売

37 東芝がELディスプレイ採用のラップトップWSを開発

38 第１回〝Digital World〟開催

41 富士通がガリウムひ素VLSIを開発
オムロンがファジイ推論プロセッサを開発
日本電気が33MHzのRISCプロセッサを発売

42 NewsFlash

46 Hewlett-Packard社Product Manager
Tad Furutsukiインタビュー

48 USENIX SUMMER 1990
特別寄稿 Technical Conference Report
砂原秀樹

54 Event Calendar

36 Sunが小型・低価格の
ワークステーションを発売

日本電気が低価格の日本語PostScriptプリンタなどを発売

日本電気㈱および日本電気ホームエレクトロニクス㈱は，日本語PostScriptを搭載したページプリンタ「PC-PR 3000 PS」，および独自のアウトラインフォントを内蔵したページプリンタ「同 1000」「同 2000」の計3機種を開発，発売した．同時に，PC-9800 シリーズ用の周辺機器やソフトウェアなども発表した．

低価格を実現したPSプリンタ「PC-PR 3000 PS」

PC-PR 3000 PSは，現行の同 602 PSの後継機に位置付けられる製品．主な特徴は，①解像度は 320 dpi，②B 4／A 4／B 5 判カット紙のほか，ハガキやOHPに印字が可能，③最大印字速度は，B 4 判で 6.7 枚／分，A 4 判で 8 枚／分を実現，④明朝およびゴシックのアウトラインフォント 8000 文字ずつを搭載，⑤同 201 シリーズおよび同 602 のエミュレーション機能を装備――など．

同 1000 は，現行の同 602 Rの下位に位置付けられる製品で，主な特徴は，①解像度は 240 dpi，②A 4／A 5／B 5 判カット紙のほか，ハガキやOHPに印字が可能，③最大印刷速度は，A 4 判で 6 枚／分，B 5 判で 6.5 枚／分を実現，④同 201 シリーズおよび 602 のコマンド体系を吸収・拡張し，イメージやグラフィックスを印字する機能を装備した独自のページ記述言語「NPDL（NEC Printer Description Language)」を搭載，⑤明朝体のアウトラインフォントを内蔵した「明朝体アウトラインフォントカートリッジ」を標準装備，⑥同 201 シリーズのエミュレーション機能を装備――など．

同 2000 は，現行の同 602 Rの後継機に位置付けられる製品で，主な特徴は，①解像度は 240 dpi，②B 4／A 4／B 5／A 5 判カット紙のほか，ハガキやOHPに印字が可能，③最大印刷速度は，B 4 判で 6.7 枚／分，A 4 判で 8 枚／分を実現，④独自のページ記述言語NPDLを搭載，⑤セントロニクスのインターフェイスを前面と背面の 2 カ所に装備，⑥明朝体アウトラインフォントカートリッジを標準装備し，オプションの「ゴシック体アウトラインフォント」との併用も可能――など．

表1 主な製品の価格

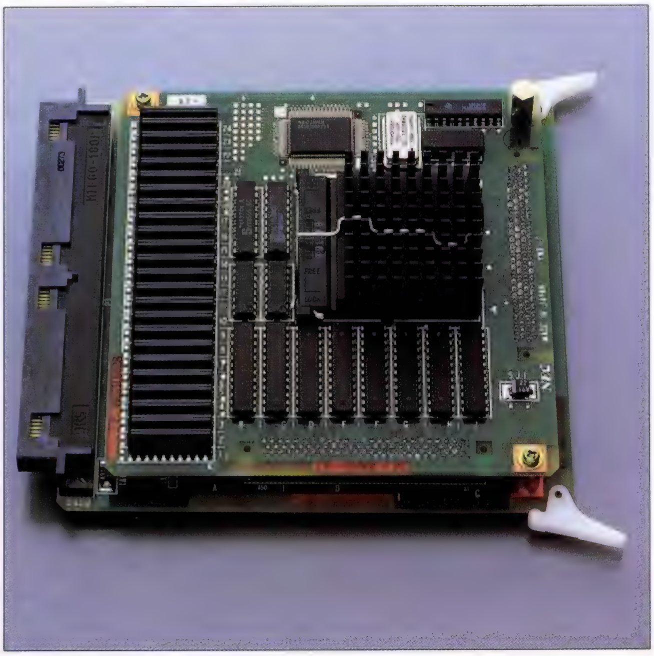
プリンタ	
PC-PR1000	28万8000円
PC-PR2000	47万8000円
PC-PR3000PS	65万8000円
周辺機器	
PC-CD103	13万5000円
PC-9875	9万8000円
PC-9873L	5万円
ソフトウェア	
PC-UX／V860ボードサポートソフトウェア	4万円
PC-UX／V860ボード サポートソフトウェア開発支援ツール	20万円
PC-UX／V860ボードGNU Cコンパイラ	1万5000円
日本語MS-DOS Ver.3.3C	
基本機能セット	9300～9800円
拡張機能セット	7700～8200円
日本語OS/2 Ver.1.1A基本セット	6万2000円

65 万 8000 円という低価格を実現した日本語PostScriptプリンタ「PC-PR 3000 PS」．

アクセス時間を向上させたCD-ROMドライブなども同時発売

同時に発表された周辺機器およびソフトウェアは，①アクセス時間を 450 msから 350 msに高速化したCD-ROMユニット「PC-CD 103」，②マ

クロック周波数 33 MHzのB60 を搭載した拡張ボード「860 ボード（PC-H 98-B 06)」．



＊1　昨年秋に発表された，クロック周波数33 MHzのi860 および 8 Mbytesメモリを搭載した拡張ボード．バスマスタ機能をサポートし，NESAバスで使用できる．拡張スロット 2 個を占有する．価格は 98 万円．

ウスとしても使用可能なA 4 判タブレット「同 9875」，③CRTディスプレイ前面に取り付けて，指で押すことによりデータを入力できるタッチスクリーン「同 9873 L」，④PC-UX／Ｖのもとで 860 ボード＊1を使用するための「PC-UX／Ｖ(Rel.3.0 B)860 ボードサポートソフトウェア」，⑤PC-UX／Ｖのもとで 860 ボード用のアプリケーションを開発するための「PC-UX／Ｖ(Rel.3.0 B)860 ボードサポートソフトウェア開発支援ツール」，⑥日本語入力の操作性改善や光ディスク装置に対応した「日本語MS-DOS Ver.3.3 C」，⑦日本語入力の操作性改善や光ディスク装置に対応した「日本語OS/２ Ver.1.1 A」――など．

価格は**表 1** を参照のこと．

表2　プリンタの主な仕様

	PC-PR1000	PC-PR2000	PC-PR3000PS
印刷方式	半導体レーザーによる レーザービーム乾式電子写真方式		LEDによる電子写真方式
ページ記述言語	NPDL		PostScript
印刷速度	Ａ４：６枚／分	Ａ４：８枚／分 Ｂ４：6.7枚／分	Ａ４：８枚／分 Ｂ４：6.7枚／分
メモリ容量	2 Mbytes		4 Mbytes
インターフェイス	セントロニクス	セントロニクス× ２（前面・背面）	セントロニクスRS-232C
用紙サイズ	A4～A5カット紙，ハガキ	B4～A5カット紙，ハガキ	B4～B5カット紙，ハガキ
外形サイズ(W×D×H)	458×368×216mm	460×499×277mm	470×525×273mm
重量	約19kg	約20kg	約30kg

AJがMacintosh用の低価格レーザープリンタを発売

アップルコンピュータジャパン㈱は，Macintosh用のレーザープリンタLaserWriterシリーズに新ラインアップ「Personal Laser WriterSC」および「同NT」の２機種を投入した．

Personal LaserWriterSCは，MacintoshのQuickDrawグラフィックスルーチンをサポートしたレーザープリンタ．主な特徴は，①解像度は300dpiを実現，②印字速度は４枚／分を実現，③CPUにはクロック周波数12MHzの68000を搭載，④メインメモリは１Mbytesを装備，⑤９～24ポイントのTimes／Helvetica／Courier／Symbol書体をディスクで同梱，⑥インターフェイスにはSCSIを採用，⑦コントローラボードの交換で，同NTへのアップグレードが可能，⑧用紙は，U.S.letter／U.S.legal／A４／B５が使用可能――など．

同NTは，PostScriptを搭載したレーザープリンタ．主な特徴は，①解像度は300dpiを実現，②印字速度は４枚／分を実現，③CPUにはクロック周波数12MHzの68000を搭載，④メインメモリは２Mbytesを装備し，最大８Mbytesまで拡張可能，⑤英字12書体37タイプフェースを内蔵，⑥HP LaserJetおよびDiablo630のエミュレーション機能を装備し，PostScript以外の文書の出力も可能，⑦インターフェイスにはAppleTalkおよびRS-232C／422を採用，⑧用紙は，U.S.letter／U.S.legal／A４／B５が使用可能――など．

両機種とも，エンジン部はキヤノン製のものを使用している．サイズは380（W）×464（D）×248（H）mm，重量は15kg．

価格は，同SCが36万8000円，同NTが54万8000円，アップグレードキットが29万8000円．

キヤノン製のプリンタエンジンを使用した低価格レーザープリンタ「Personal LaserWriter シリーズ」．

Sunが小型・低価格のワークステーションを発売

日本サン・マイクロシステムズ㈱は，カラーのエントリモデルのワークステーション「SPARCstation IPC」を開発，販売を開始した.

同マシンは，5月に販売を開始したSPARCstation 1＋と同等の性能を244（W）×264（D）×117（H）mmのコンパクトな本体に収めたものである．SPARCstation 1＋との主な違いは，メインメモリの最大容量が40Mbytesから24Mbytesと少なくなっていること，SBusスロットが3個から2個になっていることなどである.

SPARCstation IPCの発売により，カラーシステムのエントリ価格は従来の298万5000円から約20%引き下げられたことになる．同マシンは，ビジネス分野をはじめとして，CAD／CAM／CAEや画像処理などのカラーシステムの普及が見込まれる分野をターゲットとしている.

SPARCstation 1＋をさらに小型化したSPARCstation IPC

CPUは，クロック周波数25MHzのSPARC IU，数値演算コプロセッサは，同25MHzのSPARC FPUをそれぞれ搭載し，処理速度15.8MIPS（Dhrystone値），1.7MFLOPS（Linpack値）を実現している.

メインメモリは8 Mbytesを標準搭載し，4 MbytesのSIMM（Single In-line Memory Module）を4枚追加することにより，最大24Mbytesまで拡張可能である.

ハードディスクは最大207Mbytesまで内蔵可能で，SCSIを介して最大2.6Gbytesまで拡張できる．また，IBM PC互換の容量1.44Mbytesフォーマットの3.5インチフロッピーディスクも内蔵可能である．そのほか，CD-ROMや150Mbytesのカートリッジテープなども接続できる.

写真 SPARCstation IPCの本体は，幅，奥行きともSun純正の外付けハードディスクと同じサイズであり，高さは1.5台分である.

インターフェイスとして，Ethernet，SCSI，RS-423×2（RS-232C互換），オーディオI／Oポートをそれぞれ標準装備している.

拡張スロットとして，SBusを2個標準装備しているため，各種インターフェイスボードや2nd Ethernetボードなども装着可能である.

カラーディスプレイは，16インチの66Hzノンインタレースで，解像度は1,152×900ドット，1,670万色中256色の同時表示が可能である．また，同じ解像度を持つ17インチまたは19インチのモノクロディスプレイも利用できる.

表 SPARCstationIPCの仕様

	SPARCstation IPC	SPARCstation 1＋
CPU	SPARC IU（クロック周波数25MHz）	
数値演算コプロセッサ	SPARC FPU（クロック周波数25MHz）	
MIPS値	15.8MIPS	
メインメモリ	8 Mbytes（最大24Mbytes）	8 Mbytes（最大40Mbytes）
フロッピーディスク	3.5インチ（1.44Mbytes）	
ハードディスク	最大2.6Gbytes（内蔵最大207Mbytes）	最大2.6Gbytes（内蔵最大208Mbytes）
ディスプレイ	1,152×900ドット	
標準インターフェイス	Ethernet，SCSI，RS-423×2，オーディオ	Ethernet，SCSI，RS-423×2，オーディオ
拡張スロット	SBus×2	SBus×3
標準ソフトウェア	SunOS4.0以上	
サイズ	244（W）×264（D）×117（H）mm	409（W）×409（D）×71（H）mm
重量	5.5kg	11 kg
消費電力	65W	85W
価格標準構成価格	214万円（カラー）～	192万円（モノクロ）～

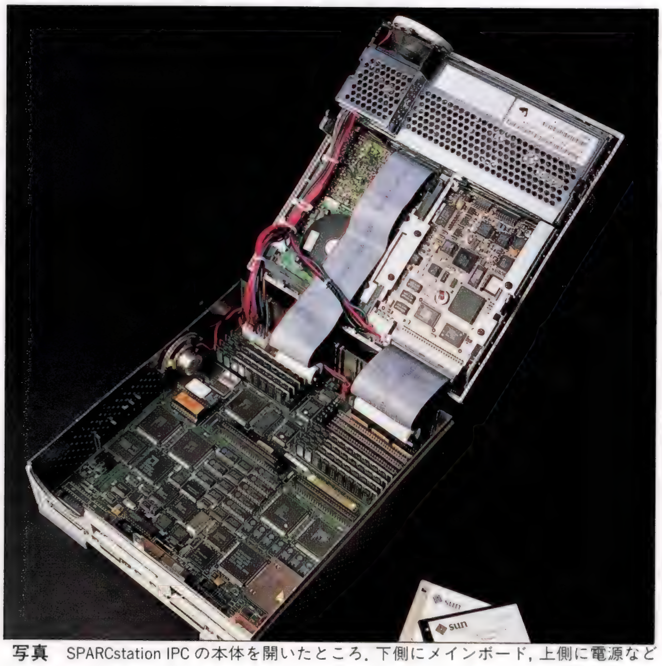


写真 SPARCstation IPC の本体を開いたところ．下側にメインボード，上側に電源などが収まっている．

OpenWindows の Release2.0をバンドル

OS は，UNIX SystemV Release 4 とソースコードレベルで互換性を持つ SunOS4.1 を採用しており，2,100種類以上ある SPARC 用アプリケーション（SPARCware）が利用可能である．

また，グラフィカルユーザーインターフェイスに OPEN LOOK を採用したOpenWindows Release2.0を標準装備している．Open Windows Release2.0 は，ウィンドウマネージャやウィンドウコンポーネントを 3 次元表示し，画面表示を見やすくしたのが特徴である．ウィンドウベースのユーティリティは，従来からの電子メール，テキストエディタ，ファイル管理ツールに，カレンダ，データベースなどを追加した．X Window Version11 Release 4 と互換性を持ち，AT&T 社の Xt＋ツールキットをベースとした OLIT (OPEN LOOK Intrinsics Toolkit) のサポートにより移植性の向上も図られている．

価格は，カラーの標準構成（本体，207Mbytes ハードディスク，キーボード，マウス，16インチカラーディスプレイ，SunOS，シングルユーザーライセンス料）で237万円．モノクロの標準構成（本体，207Mbytes ハードディスク，キーボード，マウス，17インチモノクロディスプレイ，SunOS，シングルユーザーライセンス料）で210万5000円となっている．

■問い合わせ先：〒102 東京都千代田区二番町11-19
興和二番町ビル
☎03-221-7021

東芝がELディスプレイ採用のラップトップWSを開発

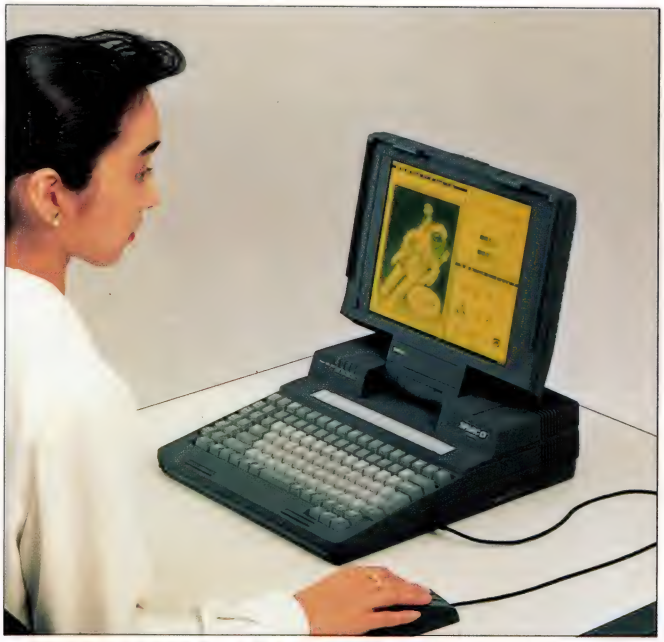
㈱東芝は，EL (Electro Luminescence) ディスプレイを搭載したラップトップ型ワークステーション「SPARC LT（AS1000／E10)」を開発，9 月より販売を開始する．

今年 5 月に発表した SPARC LT（AS1000／L10）はディスプレイ部が12インチのアクティブマトリクス方式バックライト付き液晶ディスプレイであったが，今回発表のマシンはこれを11.5インチの EL ディスプレイに変更したもの．EL ディスプレイは液晶ディスプレイに比べて応答速度が速く，2 ms を実現した（従来モデルは45ms)．表示色はアンバー（黄橙色）．

主な仕様は，①クロック周波数20MHz の SPARC チップを採用して，13.2MIPS を実現，②メインメモリは 8 Mbytes を標準装備し，最大40Mbytes まで拡張可能，③ディスプレイの解像度は1,152×900ドット，④容量180Mbytes のハードディスクを内蔵し，外部に最大 2.8Gbytes まで拡張可能，⑤ OS には SunOS をベースに開発した日英両用の OS ／ AS を採用，⑥サイズは340(W)×375(D)×99（H）mm，⑦重量は7.99kg――など．

価格は198万円．同社では，液晶モデルと EL モデルの両機種を併せて販売していく考え．

■問い合わせ先：〒105-01 東京都港区芝浦1-1-1
☎03-457-2722



EL ディスプレイを採用して，応答速度2ms を実現した「SPARC LT（AS 1000／E 10)」．

緊急レポート

第1回 "Digital World"開催

――マルチメディアの行方を探る――

写真1　オープニングセッションで冒頭挨拶をするJonathan Seybold氏

"Digital World"という世界初のイベントが6月26日から28日の3日間，米国ロサンゼルスビバリーヒルズの高級住宅街にあるホテル，ビバリーヒルトンで開催された．

このイベントは，これまで，数年先に一般化するテクノロジーに照準を当てて，まずセミナーまたはカンファレンスを開き，ついでその情報誌を出版するというビジネスを成功させているJ. Seybold氏が主催したマルチメディアカンファレンスである．

レジスト料金が895ドルと高額なのと，"マルチメディア"という概念がまだ定義づけられていないことなどもあって，参加者は425名であった．そのうち日本人は十数名しかおらず，主催のSeybold Seminarsは日本語同時通訳をサポートしなかった．

3日間のセッションのプログラムを囲みに示したが，Seybold氏が選んだマルチメディアのキーマンはこの3日間で，延べ44人である．これらの各セッションからいくつか話題性のあるものをEP newsのページ（P.174）で紹介し，ここでは，講演の中で触れられた発表およびプレス発表を中心にお伝えする．

写真2　写真左は，オープニングセッションでのSteven Jobs氏．Pixar社のCGを見ながら語っている．写真右は，Seybold氏（左）との一問一答の様子．

Digital Worldプログラム

第1日目　"Gating Technology"
Opening I, II／Digital Video／Coping with Data：Digital Telecommunications, Information Access／Evening Session

第2日目　"Gating Market"
Education／Business Applications／Multimedia：Business or Buzzword？／Publishing ： On-line Information Services,
Documentation／Entertainment

第3日目　"Application"
Computer Generated Images／Content ： Intellectual Property Computer vs. Consumer Electronics／Wrap-up

なお，セミナーのほかデモ会場もあったが，ごく小規模のもので，出展はAppleおよびNeXTのサードパーティ，Radius，Sun Microsystemsなどの各社である．

マルチメディア立ち上げの会議をMacroMind社が呼びかけ

2日目の「マルチメディアはビジネスかバズワードか？」でスピーチを行なったMacroMind社のM.Canter氏は，マルチメディア関連企業9社に呼びかけて，スタンダードについての会議を開く予定である，と発表した．

EFI社「カラーポータビリティ」がキヤノンのコピアに

EFI社E.Arazi社長は，デバイス・インディペンデントのデスクトップ・カラーパブリッシングを実現するテクノロジーとして構想を発表していた「カラーポータビリティ」について，今年12月にソースコードで出荷することを発表した．また，おそらく日本での最初のサポートとしてキヤノンのカラーコピアに採用されることを明らかにした．

ちなみに，今年4月のCEPS JAPAN'90において，E.Arazi社長が来日して講演した際には，この「カラーポータビリティ」はまだ構想段階であった．

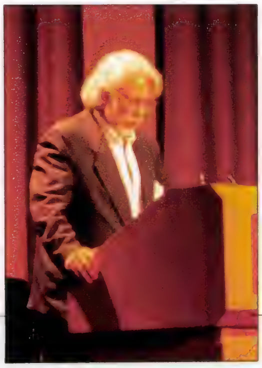


写真3　オープニングセッションで講演するStan Cornyn氏（Warner New Media社長）．

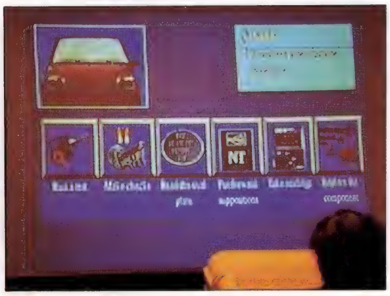


写真4　Stan Cornyn氏により紹介された「Magillar」システムによる画面デモ．

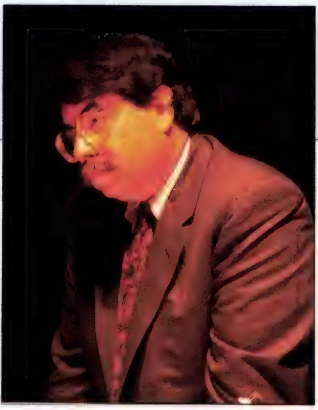


写真5　CD-ROMについて，なんとかしないと永久にビジネスにならない，と語ったDick Brass氏（Oracle Data Publishing社長）．

写真6　初日のハイライトの一人Theodor Nelson氏（Autodesk社）．

T.NelsonがXanadu/Server発売を発表

T.Nelson氏は，氏が1960年代より構想をかかげてきたXanadu/Serverを1990年末に発売する予定であることを発表した．これは，いわゆるハイパーメディア用エンジンとして設計された“革命的”ツールとなるものと語っている．

AppleのHyperCard2.0発表

Digital Worldの初日，Appleがランチタイムを利用して，突然HyperCard2.0のデモンストレーションを行なった．

いかにもAppleらしい楽しいストーリーを持たせたデモンストレーションで，HyperCard2.0でもたらされる可能性をオーバーに表現していた．HyperCard2.0の特徴などについては，本誌前号で紹介済みなので省略するが，Bantam Computer Booksの“Cooking with HyperTalk2.0”に付録のフロッピーディスクに収められている画面を示す．

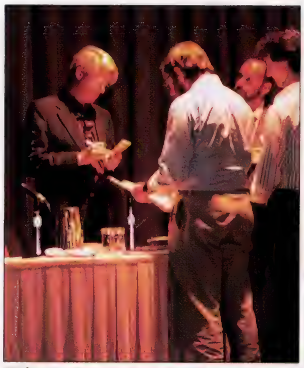


写真7　講演の直後，午後10時をまわっていたにもかかわらず，大勢の質問者（あるいは信奉者）がT.Nelson氏を取り囲んだ．

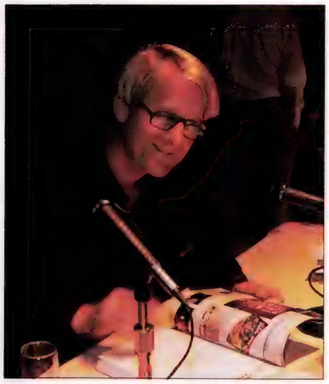


写真8　1人ずつに対してにこやかに答えるT.Nelson氏．

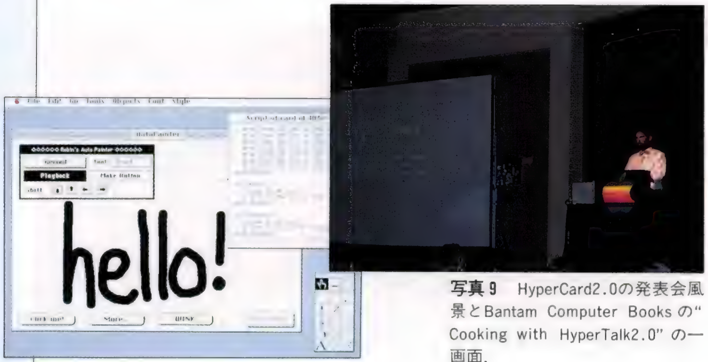


写真9　HyperCard2.0の発表会風景とBantam Computer Books の“Cooking with HyperTalk2.0”の一画面．

写真10　“Education”についてのパネルディスカッション
右からRobert Abel（AND Communications），Ms.Lucy（IBM），Robert Winter（UCLA音楽科），James Catterall（UCLA教育学），Hellen Kelly（Los Angels Unified Sch. District），Seyboldの各氏．

RadiusとMacroMindとABC NEWSの共同発表

初日のAppleの発表に続いて，Radiusも自社ブースでプレスカンファレンスを行なうとアナウンスした．「Radius TV」という新製品の発表であった．発表には，RadiusのMichael Boich社長，MacroMindのJohn Scull社長，ABC NEWS InterActiveおよびThe Post Groupの各重役が顔を揃えた．

「Radius TV」は，オーディオおよびテレビの信号をMac環境でリアルタイムに再生するコンバータで，16ビットのビデオイメージ（30フレーム／sec）をリアルタイムで640×480ピクセルのウィンドウ上に再現できる．また，サウンドもリアルタイムでデジタル化するなど，テレビに含まれるあらゆる情報を，Macによりアクセスしたりコントロールすることが可能になる．

なお，これはビデオディスク，RGBおよびスチルビデオカメラ，ビデオレーザディスクのいずれにも対応できる．

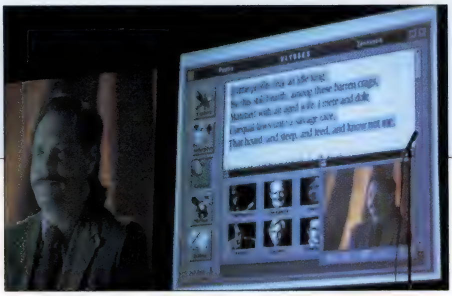


写真11
R.Abelのプレゼンテーション

・システム構成例

MacintoshII, Radius TV, Radiusカラーディスプレイ(Radius以外のディスプレイでも，またモノクロでも可),「Director」(オーサリングソフト)，ABC NEWS社のデータ.

たとえば，上記のような組み合わせで，ABC NEWS社のテレビ放送をDirectorにより，あるアプリケーションとする．Radius TVは，Macのスロットに差し込むことも外付けでも対応できる．これをTVチューナーに接続する．

写真12 コンピュータによるカラーイメージ処理について講演するEfraim Arazi氏(Electronics For Imaging社長).

・価格と出荷

2750ドルで，7月末出荷と発表.

・サードパーティ

MacroMind, ABC NEWS InterActive, The Post Groupの各社はRadiusTVをサポートするアプリケーションを開発することを表明した．Directorはβサイトでの開発アプリケーションをデモ

写真13 著作権と知的所有権についてのパネルディスカッション．右からRobert Stein(The Voyger CO .)，David Bohman（ABC NEWS InterActive)，Scott Mize（Apple Computer)，Steven Holtzman(Radius) David Nimmer(Irell & Manella）の各氏.

写真14 「RadiusTV」の記者発表会.

写真16 Wrap-up（まとめ）でのパネリスト.
右からPaul Brainerd（Aldus Corp.社長)，David Liddle（Metaphor Computer System)，Bernard Peuto（Sun Microsystems)，Seyboldの各氏.

写真15 講演中のDavid Bohman氏.

「Magillar」システム構想

Warner New Media社のS.Cornyn社長は，実現化をめざす新製品として，「Magillar」システムを紹介した．これは，“テレビとコンピュータとのハーモニー”と表現していたが，TVモニタとLDプレイヤーとコンピュータとを組み合わせてインタラクティブな利用ができるようにする，というもの.

いく通りかのシステム構成があるのだが，一番ターゲットとしているのは，各家庭のテレビをケーブルによってホストコンピュータに接続し，インタラクティブなインターフェイスを提供するシステム．まだコードネームの段階だが，'91年には発売の予定.

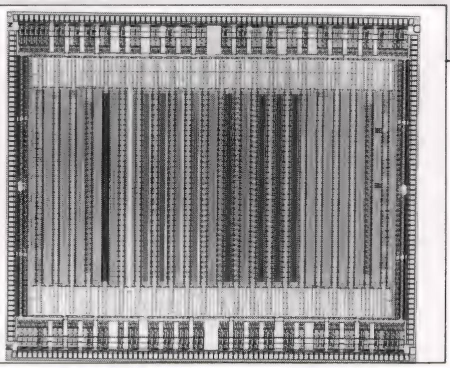
写真17 Brainerd社長とSeybold氏との質疑応答の様子.

＊1　ガリウムとひ素の化合物ウェハを基板材料としたゲートアレイ．ゲートアレイは基板上にあらかじめトランジスタや抵抗を作っておき，その上にユーザーが電気回路を設計配線し，集積回路を構築するもの．

＊2　Emitter Coupled Logic．バイポーラロジックの一種．市販ロジックICとしては現在最高速．電流切替型論理回路の出力に，バッファ回路としてエミッタ・フォロワを付加したもの．

＊3　Fuzzy Logic Interface per Second．1条件1結論ルールに換算した推論速度で，1秒間に推論できる回数を指す．

富士通がガリウムひ素VLSIを開発



ECLゲートアレイの約1/3という低消費電力化を実現したガリウムひ素ゲートアレイ「MB 53000 シリーズ」．

富士通㈱は，米Vitesse Semiconductor社の技術協力により，3,000～30,000 ゲートのガリウムひ素ゲートアレイ*1「MB 53000 シリーズ」6製品を開発，10月より受注生産を開始する．総合半導体メーカーとしてガリウムひ素VLSIを販売するのは，世界で初めて．

同シリーズは，Vitesse社のFURYシリーズと完全互換性を持ったゲートアレイ．スピードとゲート規模は現在最先端のシリコンECLゲートアレイ*2と同等（ゲート遅延時間 80 ps）ながら，ゲート当たり 0.4 mWとECLの約1/3という低消費電力化を実現したのが最大の特徴．他の主な特徴は，① 0.8 μmのプロセス技術を採用，②フリップフロップのトグル周波数は1 GHz以上，③入出力I/Oインターフェイスや電源，パッケージなどの仕様はシリコンICと互換性を実現，④パッケージはQFPとPGAを用意——など．

価格は，3,000 ゲート規模のMB 53030 が単価3万円(開発費 570 万円)，30,000 ゲート規模の同 53150 が単価 16 万円(開発費 1500 万円）などとなっている．

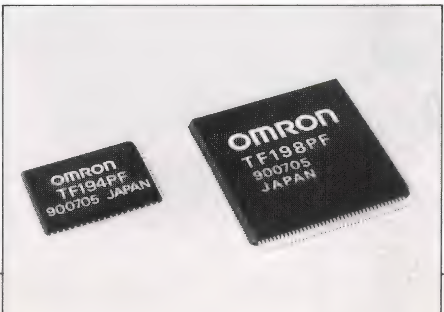
オムロンがファジイ推論プロセッサを開発

オムロン㈱は，デジタルファジイプロセッサ「DFP・マルチタスクシリーズ」および「同コントローラシリーズ」の2シリーズ8製品を開発，11月より販売を開始する．

マルチタスクシリーズは，高速・大量データ処理用途向けの製品．主な特徴は，①汎用DFPとしては世界最高速の 10 MFLIPS*3の演算速度を実現，②最大 32000 ルールを 128 グループ化して取り扱うことが可能，③ 1600 万種のメンバーシップ関数を使用可能——など．

コントローラシリーズは，各種コントローラや小型機器組み込み用途向けの製品．主な特徴は，①現行のファジイチップFZ-3000／3010 に使用されているDSPに比べて推論速度を2倍に向上し，0.2 MFLIPSを実現，②制御向きの重心法と判断・認識向きの最大高さ法の確定演算機能をサポート可能，③複数のルールを3グループ化し，制御目標に応じて切り替えが可能——など．

サンプル価格は，マルチタスクシリーズが2万～3万円，コントローラシリーズが 8000～1万円．量産価格は，マルチタスクシリーズが 4000～1万円，コントローラシリーズが 1800～2500 円．



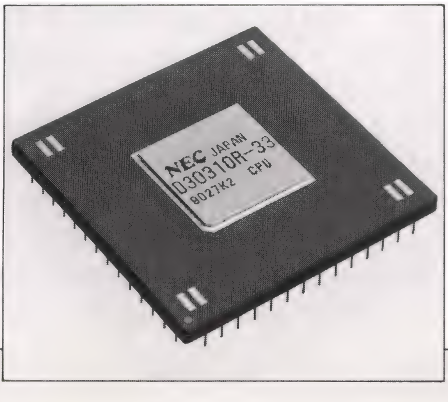
ファジイ演算速度10 MFLIPSと，汎用マイコンの約10 000倍の性能を実現した「DFP・マルチタスクシリーズ」．

日本電気が 33 MHzの RISCプロセッサを発売

日本電気㈱は，クロック 33 MHzの 32 bitRISCプロセッサ「VR 3000 A (μPD 30310)」を開発，サンプル出荷を開始した．同プロセッサは，米MIPS Computer Systems社との技術提携により開発したもの．

同プロセッサの主な特徴は，① 0.8 μmのCMOSアルミ2層プロセス技術を採用し，6.43×7.35 mmのチップ上に 115,000 トランジスタを集積，②処理性能は，最大 33 MIPS，平均 26.4 MIPSを実現，③ソフトウェア・ハードウェアとも現行のVR 3000 との上位互換性を確保，④アドレスバスの3ステート処理により，マルチプロセッサによるシステム構築が可能，⑤VR 3000 に比べて約30 %の低消費電力化を実現，⑥パッケージは 175 ピンPGA，⑦OSとしてはUNIX System V Release 3 に 4.3 BSDを統合したRISC／OSを用意，⑧数値演算コプロセッサとして 9.2 MFLOPSのVR 3010 Aを用意——など．

サンプル価格は7万円



0.8 μmCMOSプロセス技術を採用して，11 5000 個のトランジスタを集積したRISCプロセッサ「VR 3000 A」．

MitacがWindows3.0をバンドルしたIBM PC互換機を発売

㈱マイタックジャパン（☎03-586-1631）は，Windows3.0を標準バンドルしたIBM PC互換機「MPC2386E」を発売した．主な特徴は，①CPUにはクロック周波数20MHzの386SXを採用，②メインメモリは1Mbytesを標準装備し，メインボード上で最大8Mbytesまで拡張可能，③5インチと3.5インチのFDD各1個を装備，④グラフィックカードとして解像度1024×768ドットのEnhanced VGAを標準装備，⑤拡張スロットは6個装備，⑥ソフトウェアとして，Windows3.0のほかMS-DOS Ver.4.01，GW-BASIC Ver.3.2，ディスプレイドライバなどを標準装備――など．価格は42万8000円．

マイクロウェアがPC-9800用のOS-9000を発売

マイクロウェア・システムズ㈱（☎03-257-9000）は，マルチタスク・マルチユーザーのPC-9800シリーズ用OS「OS-9000／PCデベロップメント・パッケージ」を発売した．パッケージには，システム本体，約70種のユーティリティコマンド，Cコンパイラ，システムデバッガ，Cソースレベルデバッガ，漢字対応スクリーンエディタ（uMacs），日本語入力FP（VJE-γ）が含まれる．主な特徴は，①OS-9との上位互換を実現，②完全にプロテクトモード上で動作，③最大ファイル長は4.2Gbytesまで可能，④最大ディスク容量は128Tbytesまで可能――など．対応機種は，PC-9801RA／RS／ES．価格は19万円．

マイクロソフトがPC-9800用統合ソフトを開発

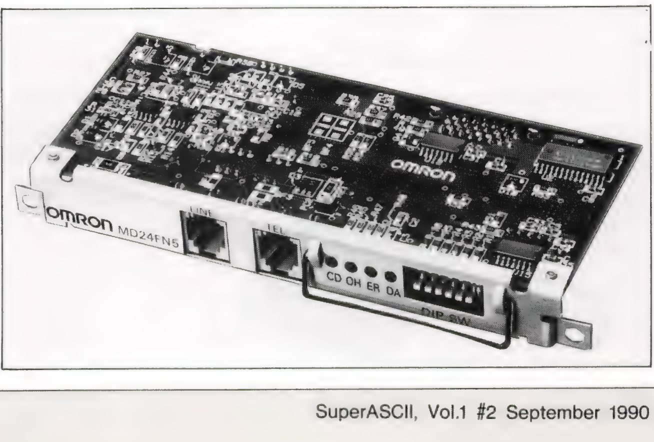
マイクロソフト㈱は，PC-9800シリーズ用の統合ソフト「Works Ver.2.1」を開発，9月より販売を開始する．同ソフトは，ワープロ，表計算，データベース，通信の4つの機能を統合化したもの．主な特徴は，①Windowsライクな独自のウィンドウ環境を装備し，最大8個までのウィンドウを同時にオープン可能，②ワープロは，全角／半角／横倍角などの文字種や強調／網掛け／下線などの文字修飾が可能，③表計算は，4096行×256列までの取り扱いができ，棒／帯／折れ線／円など9種類のグラフ表示が可能，④データベースは，最大レコード数が4096個，最大フィールド長が255桁，最大フィールド数が256項目，⑤通信は，最大9600bpsまでの通信速度に追随し，ファイル転送プロトコルとしてX MODEMを装備，⑥約220項目のオンラインヘルプを装備，⑦対話式で操作を学習できる学習ソフトを添付――など．動作環境は，MS-DOS Ver.3.1以上，メインメモリ640Kbytes以上が必要．価格は4万円．

RasterOpsがPS/2用のフルカラーボードを発売

米RasterOps社は，IBM PS/2用の24bitフルカラーボード「1024MC」を発売した．同ボードは，解像度640×480ドットで1670万色の同時表示が可能なもの．Windows3.0およびPresentation Manager用のグラフィックドライバを標準装備している．現在のところ，PageMakerやPowerPoint，Corel Drawなどの約20種の主要ソフトウェアをサポートしている．価格は4395ドル．

オムロンが2400bpsモデム6機種を発売

オムロン㈱は，エラー訂正プロトコルにMNPとV.42を搭載した2400bpsモデム「MD24シリーズ」6機種を発売した．内訳は，①MNPのクラス5を搭載したポケットタイプ「MD24FP5 II」，②同クラス4を搭載したポケットタイプ「同FP4 II」，③同クラス5を搭載したPC-9801N内蔵タイプ「同FN5」，④同クラス4を搭載したPC-9801N内蔵タイプ「同FN4」，③同クラス5を搭載したJ-3100SS内蔵タイプ「同FJ5」，④同クラス4を搭載したJ-3100SS内蔵タイプ「同FJ4」――の6機種．主な特徴は，①制御コマンドは，Hayes社のATコマンドとCCITT V.25bisに準拠，②通信規格は，CCITT V.22bis／V.22／V.21およびBell 212A／103に対応，③通信中以外は自動的に消費電力を抑制するパワーセーブモード機能を搭載――など．価格は，同FP5 IIが4万2800円，同FP4 II 3万8800円，同FN5が4万4800円，同FN4が3万9800円，同FJ5が4万9800円，同FJ4が4万4800円．



*1不揮発性で電気的にリード・ライトが可能なメモリ．書き込まれた情報に瞬時にアクセスできるのが特徴．

インテルが2Mbitフラッシュメモリを発売

インテルジャパン㈱（☎0298-47-8511）は，業界で初めて容量2Mbitのフラッシュメモリ*1「28F020」を開発，サンプル出荷を開始した．主な特徴は，①最大リードアクセス時間は150ns，②1byte当たりの書き込み時間は10μs，③消去時間は1チップ当たり2秒，④パッケージはPLCCとJEDEC規格準拠のPDIP，今回新たに開発したEIAJ規格準拠のTSOPの3種を用意——など．サンプル価格は，PLCC／PDIP版が7000円，TSOP版が9000円．

日商岩井がSGML構造化エディタを発売

日商岩井㈱（☎03-588-4598）は，英Yard Software Systems社が開発したIBM PC用のSGML構造化エディタ「Write-IT」の国内販売を開始した．同エディタは，すでに販売しているSGMLパーサMark-ITとのリンクが可能なもので，TeXなどとリンクさせることで，SGML統合システムが構築できる．同エディタを使えば，テキストを入力する場合にコンパイルされたDTDを選択することで編集中の位置を構造的に確認が可能．価格は15万円．

リードレックスがDTPソフトのSun対応版を発売

㈱リードレックス（☎03-797-3911）は，Sunシリーズ対応のDTPソフトウェア「PRESBOX Sun対応版」を発売した．主な特徴は，①SunViewに対応し，WYSIWYGを実現，②文字入力・編集や図形描画，イメージ入力，作表，レイアウト設定などの文書作成機能を装備，③縦・横書き日本語のジャスティフィケーションやルビ打ち，重ね書き，組み文字などの日本語編集機能を装備，④HP-GL形式のCADデータやSunViewのビットイメージ，UNIX標準テキストファイル，一太郎文書ファイル，花子・Z's STAFF Kidのイメージファイル，MultiplanなどのSYLKファイル，カンマ区切りのデータベースファイルなどの取り込みが可能，⑤明朝体とゴシック体のアウトラインフォントを装備し，文字サイズは6～72ポイントまで0.5ポイントきざみで指定が可能，⑥日本語PostScriptや各種ページプリンタのネイティブモードに対応——など．価格は48万円．

ソニー・テクトロニクスが低価格カラーPSプリンタを発売

ソニー・テクトロニクス㈱（☎03-779-7603）は，198万円と業界で初めて200万円を切った解像度300dpiのカラーPostScript熱転写プリンタ「Phaser 4693PX」を発売した．主な特徴は，①PostScriptおよびHP-GLエミュレーションをサポート，②CPUにはクロック周波数16MHzの68020を採用，③メインメモリは6Mbytesを標準装備し，最大10Mbytesまで拡張可能，④用紙は，レター／リーガル／A4／A4特殊が使用可能，⑤インターフェイスは，AppleTalkおよびRS-232C，セントロニクスを装備，⑥PANTONEカラールックアップテーブルをサポート——など．

エプソンが20MHz386SXを搭載したラップトップを発売

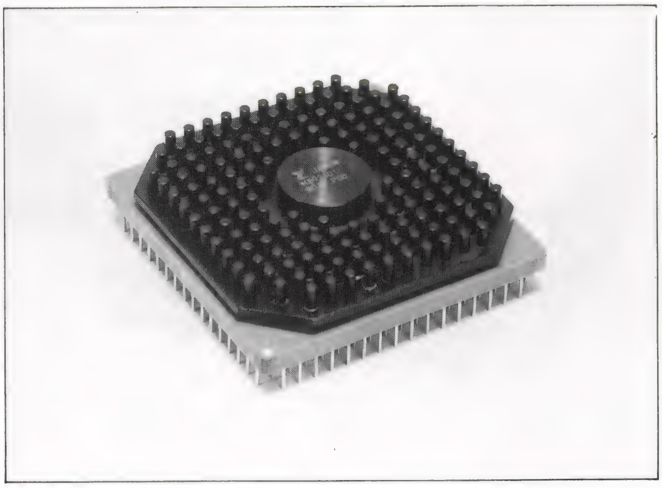
セイコーエプソン㈱は，CPUにクロック周波数20MHzの386SXを搭載したラップトップマシン「PC-386LSR」を発売した．主な特徴は，①メインメモリは1.6Mbytesを標準装備し，最大9.6Mbytesまで拡張可能，②ディスプレイには，解像度640×400ドットのバックライト付きの8階調白液晶を採用，③3.5インチFDD2台を内蔵し，20M／40Mbytesの3.5インチHDDパックを内蔵可能，④拡張スロット2個を装備——など．価格は，本体が53万8000円，20MbytesHDDパックが14万2000円，同40Mbytesが23万2000円．

Compaqが33MHzの486マシンを発売

米Compaq Computer社は，同社の最上位マシンとしてCPUにクロック周波数33MHzの80486を搭載したIBM PC互換機「Deskpro 486-33L」を発売した．同マシンは，メインメモリは4Mbytesを標準装備し，最大100Mbytesまで拡張が可能．拡張バスには，EISAを採用している．価格は，容量650MbytesのHDDを搭載したModel650が1万9499ドル，同320MbytesHDD搭載のModel320が1万6499ドル。同120MbytesHDD搭載のModel120が1万3999ドル．

富士通が最上位のTRONプロセッサなどを発売

富士通㈱は，32bitTRONプロセッサ「GMICROファミリ」の最上位機種として，「GMICRO F32／300（MB92301）」およびクロックパルスジェネレータ「MB92461」のサンプル出荷を開始した．GMICRO F32／300の主な特徴は，①1.0μmCMOSプロセス技術を使用して，約90万トランジスタを集積，②整数演算能力は現行の同F32／200の約2倍の37,000Dhrystone，17MIPSを実現（クロック周波数25MHz動作時），③ITRON仕様リアルタイムOSのREALOS／F32とUNIX System V Release3.1などをサポート，④現行の数値演算コプロセッサMB92811，DMAコントローラMB92411，割り込みコントローラMB92421，タグメモリMB92431の使用が可能――など．MB92461は，MPU，周辺デバイスおよびシステム用の各種クロックを供給するもの．価格は，クロック周波数20MHzの同F32／300が16万円，MB92461が1万5000円．



神戸製鋼がMac用の大容量HDDを発売

㈱神戸製鋼所（☎03-5476-9223）は，Macintosh用の大容量ハードディスク「DATA HOUSEシリーズ」2機種を発売した．内訳は，容量320MbytesのHDDと同150Mbytesのテープストリーマを搭載した「DH-320MS」，および同660MbytesのHDDと同150Mbytesのテープストリーマを搭載した「同660MS」．HDDの平均アクセス時間は14ms，最大データ転送速度は1.9Mbytes／s．価格は同320MSが89万8000円，同660が128万円．

日商エレと富士ゼロックスがネットワークファイルサーバを発売

日商エレクトロニクス㈱と富士ゼロックス㈱は，米Auspex Systems社に資本参加し，Auspex社が開発したネットワークファイルサーバ「NS5000」の国内販売を開始した．NS5000の主な特徴は，①68020またはRISCチップ4個以上のマルチプロセッサによる機能分散を実現，②55MbpsのVME互換バスを採用，③レスポンス時間を従来製品の5～10倍に向上，④ディスク装置への並列同時読み書き機能を実現，⑤Ethernetポートは最大8個，⑥ディスク装置は最大13.3Gbytesまで拡張可能，⑦I／Oキャッシュメモリは最大96Mbytesまで搭載可能――など．価格は，エントリレベルマシンで2600万円程度．

LSIロジックがRISCチップ用の周辺チップセットを発売

LSIロジック㈱（☎03-589-2711）は，RISCプロセッサ用の周辺LSI 5製品を開発し，「MIPset M-3」として発売した．同セットは，すでに同社が発売しているRISC CPUのLR3000およびFPUの同3010用の周辺デバイス群で，内訳は，①リセット／割り込みコントローラ「LR3201」，②L-Busコントローラ「同3202」，③DRAMコントローラ「同3203」，④DRAMデータバッファ「同3204」，⑤ブロック転送バッファ「同3205」，⑥RISC CPU「同3000」，⑦RISC FPU「同3010」——となっている．クロック周波数20MHz版と25MHz版を用意し，1991年中には33MHz版の出荷も予定している．パッケージは，プラスチックQFPとセラミックPGAを用意．価格は，20MHzのプラスチックQFP版で4万8000円(CPU，FPUを除く)．

NSが業界最高速のECL PLDを発売

ナショナルセミコンダクタージャパン㈱（☎03-299-7001）は，入出力遅延時間が2nsと業界最高速を実現したECLプログラマブルロジックデバイス「PAL10016C4-2」を発売した．同チップは，同社のタングステンヒューズ技術を使用して220mAと消費電力の低減化を図り，28ピンPLCCパッケージでの供給を可能とした．スーパーミニコンピュータやワークステーションなどの高速スイッチング回路への使用が期待されている．価格は9500円(100個ロット時)．

DECがVAXファミリに小型汎用機を投入

日本ディジタル イクイップメント㈱は，入出力性能とネットワーク性能を向上した小型汎用コンピュータ「VAX4000モデル300」を開発，9月より出荷を開始する．主な特徴は，①CPUに同社開発の最新CMOS半導体チップを採用し，現行のMicroVAX3900に比べて約2倍の8MIPSを実現，②ディスクコントローラおよびEthernetコントローラに10MIPSの性能を持つRISCチップを搭載，③メインメモリは最大128Mbytesまで拡張可能，④HDDは，容量1Gbytesのものを最大28台まで接続可能，⑤現行のVMS OS用のアプリケーションの利用が可能——など．価格は，1332万円から．

AMDが最高速の2 MbitEPROMを発売

日本AMD（☎03-346-7560）は，業界最高速の2 MbitEPROM「Am27C2048」の量産を開始したと発表した．Am27C2048のアクセス時間は120ns．ほかの同社製EPROMと同様，Flashriteプログラミングアルゴリズムを使用して，チップ全体のプログラムを60秒以内で行なうことができる．パッケージは，40ピンDIPおよび44ピンLCCを用意している．価格は6850円（100個ロット時）．

日電が50MHzの68030を搭載したワークステーションを開発

日本電気㈱は，CPUにクロック周波数50MHzの68030を採用したワークステーション「EWS4800／30」を発売した．主な特徴は，①演算速度は現行機種の約1.7倍の12MIPSを実現，②現行のEW4800シリーズとオブジェクトコードレベルでの互換性を実現，③X Window Ver.11 Release 4とOSF／Motifをベースに開発された「日本語Mf」やアイコンを使った「X．desktop」などのユーザーインターフェイスを用意——など．価格は，標準構成（メインメモリ16Mbytes，328MbytesHDD，3.5インチFDD，20インチカラーディスプレイ，高速描画機構，キーボード，マウス，OS，X Window）で455万円．8月末出荷開始の予定．

＊1　グラフィックス処理のトランスフォーメイションエンジン（座標変換エンジン）からレンダリングエンジン，フレームバッファに至るパイプライン．

Hewlett-Packard（HP）社が先頃発表したグラフィックスエンジン「PersonalVRX」のプロダクトマネージャ，Tad Furutsuki氏が来日した．来日の目的は，VRXグラフィックスの発表会ならびに日本市場の調査である．今回は，グラフィックスエンジンについてお話を伺った．

☆　☆

――まず，PersonalVRX／TurboVRXでi860を採用した理由をお聞かせください．

Furutsuki：何点かありますが，まず，浮動小数点演算能力が優れていることです．あと，オンボードキャッシュが多いため，安いDRAMを外部メモリとして使うことができること，優れたコンパイラやデバッカ，アセンブラがすでにあったことなどです．

――PRISMやPA-RISCといった自社のRISCプロセッサをグラフィックスエンジンとして使うことは考えなかったのでしょうか．

Furutsuki：PRISMに関しては，DN10000VSで使用していますが，TurboVRXのプロジェクトが発足した当初は，まだApollo社と合併していなかったため考えられませんでした．また，PA-RISCに関しては，グラフィックスアクセラレータとして使うことはないと思います．というのは，PA-RISCは汎用プロッセッサとしての性格が強いため，グラフィックスコントローラといった用途には向いていないのです．それに，浮動小数点演算能力ではi860のほうがまだ高いということもあります．

――i860の持つグラフィックス命令が使いたかったということはないのでしょうか．

Furutsuki：それはありません．実際にもそれらの命令は使っていません．

――i860はグラフィックスエンジンのどこの部分を担当しているのでしょうか．

Furutsuki：主にグラフィックストランスフォーメイション（座標変換），ポリゴンの計算，タイリングに使ってます．これはグラフィックスパイプライン*[1]のグラフィックスアクセラレータ専用のものです．

――そうすると，ユーザーがi860をアクセスすることはできないわけですね．

Furutsuki：今は，Starbaseでサポートされていませんから，使うことはできません．

――PersonalVRXとTurboVRXのi860×1のバージョンの違いについてなのですが，i860のクロック周波数だけなのでしょうか．

Furutsuki：アーキテクチャはどちらも同じなのですが，TurboVRXではアンチエイリアシングやテクスチャマッピングなどをサポートしています．PersonalVRXではサポートしていません．これらは，スキャンコンバータ部分のカスタムVLSIでサポートしているのですが，このカスタムVLSIが違うのです．

――本体とグラフィックスが独立していることで，I／Oやメモリのスループットが問題になるかと思いますが，VRXではどのように解決しているのでしょうか．

Furutsuki：グラフィックスパイプラインの中では，300Mbytes／秒の速度でデータを動かすことができるので，今のところ問題となるのは，CPUとグラフィックスエンジン間のデータ転送速度です．ここは，30Mbytes／秒のスピードとなってます．アニメーションとかフレームレートの速いアプリケーションなどではボトルネックとなってしまいますが，MCADなどの用途ではまだ大丈夫です．将来的にはCPUの速度を向上させることと，その過程で新しいグラフィックスバスを採用することで改善できると思っています．

＊2　SDRCは，Structual Dynamics Research Corp.のことで，ここではSDRC社のMCAD／CAEアプリケーションのI DEASシリーズのこと．UNIGRAPHICSは，McDonnell Douglas Manufacturing & Engineering Syst.社の3次元のCAE／CAD／CAMデータベースソフト．

――PHIGSやStarbase以外のライブラリの採用の予定はあるのでしょうか．たとえば，RenderManやDoreなどは検討されたのでしょうか．

Furutsuki：RenderManについては検討しましたが，コードのサイズや速度の点などが問題になりました．それに，HPはグラフィックスリーダーであると考えていますので，そのようなソフトをわざわざライセンスする必要はないと思っています．自社で十分やれるというわけです．

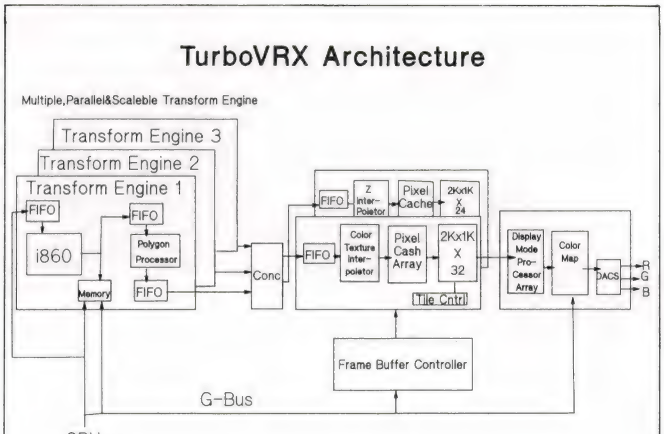
――今回の製品はどのような市場を狙っているのでしょうか．

Furutsuki：PersonalVRXに関してはMCADやMCAE用のエンジンとして，ソフトウェアではSDRC*2やUNIGRAPHICS*2を安い値段で走らせるといったシステムです．TurboVRXですが，これはハイエンドのMCAD, MCAEでコンセプシャルモデリングやビジュアリゼーション分野を狙った製品です．競合相手との位置付けですが，PersonalVRXはPersonalIRIS，TurboVRXはPower Visionに相当するマシンといえます．

――日本の市場についてはどのような印象を得ましたでしょうか．

Furutsuki：3次元の分野についてですが，日本とアメリカの設計の仕方などの違いもあるのでしょうが，ハイエンドとローエンドに分化しているという印象を受けました．これは，大きな会社で，デザインから製造，加工まで同じ3次元のCADデータベースを使って行なえるところは，下請けなどの問題もあるので日本では少ないと思います．3次元のCADデータを下請けに渡しても難しいのではないのでしょうか．

――最後に，1995年にはどのようなマシンが出てくると思われますか．



TurboVRXのアーキテクチャ概念図．PersonalVRXとは，トランスフォームエンジンが並列であることと，図の中央のASICが違う．グラフィックスエンジン内は300Mbytes／秒のデータ転送能力を持っている．

Furutsuki：5年後というのはこの分野ではどうなるかまったく分かりませんので，2年後の'92年ということにさせてください．

まず，ハイエンドについてですが，ポリゴンの描画能力が今のベクタの能力，200万～300万ポリゴン／秒ぐらいのマシンが出るのではないでしょうか．

ハイエンドのアーキテクチャというのは，グラフィックスパイプラインを持ったもので，i860や今後出てくるであろうi960とかを使ったものが多いと思います．ビジュアリゼーショングラフィックスでリアリズムに優れたもの，たとえば，リアルタイムのレイトレーシングやラジオシティといったものがサポートされるでしょう．あと，I／Oとしては，ビデオやHDTVといったマルチメディア的な端子が付くのではないでしょうか．

ローエンドについてですが，5000～7000ドルの3次元をサポートしたマシンが登場すると思います．アーキテクチャとしては，グラフィックスパイプラインでスキャンコンバータの部分だけをゲートアレイやASICに置き換えて，座標変換はCPUに任せてしまうといったものになるでしょう．

HPがこういった製品を作るというわけではなく，あくまでも，業界にこのような製品が出てくるということです．

Tad Furutsuki(古月忠兆)
グラフィックス プロダクト マーケティングが専門．PersonalVRXのマーケティングを担当．ComputerVision（現，PRIME）社，Apollo Computer社を経て，UCLAのMBA課程を終了後，'88年にHP入社．

USENIX SUMMER 1990

6月11日から5日間にわたって，カリフォルニア州アナハイムで「USENIX」が開催された．

今回の出展社数は約60社．それほど大きい数字ではない．急ぎ足で回れば1時間余りですべてを見て回れる程度の規模だ．IBM，Sun Microsystems，HP／Apollo，DEC，AT&Tなど大手は一応出揃ったが，Apple ComputerやSilicon Graphicsなどが参加しなかった．そもそもUSENIXは，どちらかというとアカデミックな話題が中心の技術コンファレンスなので，2,000名を超える参加者のほとんどはプログラミング開発，システム運営などの現場に直接携わっている人たちである．

シカゴから来たある技術者は「USENIXでは僕のシステムの持っている技術的な問題について何か得られるものがあれば」と考えて参加したという．また，コロラド州の大学のコンピュータサイエンス学科で教鞭をとっているある教授は「私と同じように教育の現場にいる人たちと直接話がしたくて参加しました」と言っていた．出展社側から見ると，直接商談ができるような相手が参加しないので，USENIXが敬遠されがちなのもうなずける．

1988年のサンフランシスコでのUSENIXでは，ちょうどOSFと

写真1　Sun Microsystems社の「SPARCstation SLC」．モニタの後ろ側にSPARCチップを搭載した基板が入っている．SLCは主にXターミナルをターゲットに開発されたマシンで価格は4000ドル台．

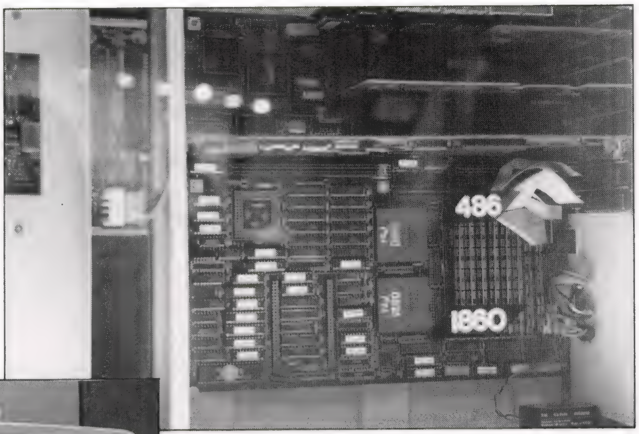


写真2　UHC社の80486／i860マシン「UH8000」．高速なグラフィックス処理を盛んにアピールした．最大80486×4個，i860×16個を搭載可能．

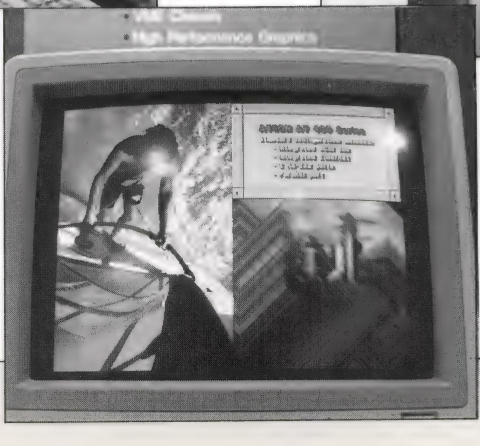


写真3　Data General社の「AViiONシリーズ」．このマシンはCPUにMotorola社の88100を採用している．

UIが火花を散らしている最中だったこともあり，各社がそれぞれの陣営のサインを表示して競い合っていたものだが，今回の展示会場ではそうした光景もまったくといってよいほど影をひそめてしまったようだ．OSFとUIのマージの話もここしばらくはなさそうだし，ユーザーインターフェイスはMotifでほぼ決まりといった情勢だからだろうか．

というわけで，コンファレンスは別として展示会のほうは多少盛り上がりに欠けるものとなってしまった．

X Windowが最大の売り物

展示会場に入ってみると，やはりX Windowが最大の売り物となっていた．X Windowが，UNIXの標準ウィンドウシステムとしてますます定着してきているというのもうなずける．NCD社のようなXターミナル専門の開発会社やPC上でXサーバを動かしているところなどが入り乱れての展示となった．Xサーバ関連製品市場はすでに飽和状態に達した感がある．こうした中で，Sun Microsystems社が先頃発表したばかりのSPARCstationSLCは，約5000ドルという低価格で注目を集めていた（**写真1**）．SLCは，Xサーバ製品に対抗するために開発されたといわれているが，ハードディスクを外付けすることにより，スタンドアローンマシンとしても使用できる．さらに7月になってSPARCstation 1＋(通称ピザボックス)の小型バージョン「SPARCstationIPC」を発表し，今後UNIXマシン本体の低価格化にさらに拍車がかかるものと思われる．

会場で見かけた最も変わったマシンは，ヒューストンに本拠をおくUHC社の「UHC8000」．**写真2**で見て分かるように，80486搭載のマザーボード上にi860までも搭載してしまっている．普通にUNIXを動かしている環境下ではi860はコプロセッサとして動作するが，同シリーズは80486なしのバージョンも用意している（現時点では，i860×1個とi860×2個のものの2種類だが，将来的にはi860×16個および80486×4個の搭載が可能になるという）．同社のUNIXはSystem V Release 4をベースにしたもの．i860専用，80486／i860ハイブリッド用のバージョンも用意している．

Data General社のAViiONはMotorola社の88100をベースにしたマシンだ．会場で88Kベースのマシンは多分これだけではなかったかと思われる．このAViiONのUNIXであるDG／UXはSystem V Release3.1をベースにしているが，Release 4の多くの機能を包含している（**写真3**）．

IBM社は，RS／6000を展示．X Window上で動作するPHIGSなどのデモを行なった（**写真4**）．320クラスのマシンでもかなり高速に動く．この程度の描画速度があれば，秒間数コマのリアルタイム表示も可能だ．IBMは，同時にあまり目立たないところでAIX版のNextStepを展示していた（**写真5**）．X Windowが優勢な展示会場内では異色なデモだった．

写真4　IBM社の「RS／6000」．PhigsによるX Window上での3D処理の例．数コマ／秒のリアルタイム表示も可能だ．

写真6　NeXT社のプライベートルームにて．NeXTマシンでもInformix社のWingzが走るようになった．

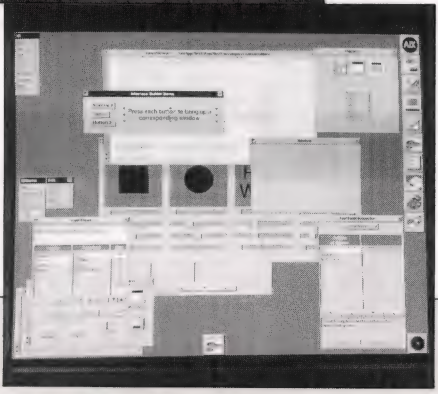


写真5　IBM社のRS／6000のAIX上で動作する「NextStep」．右上のロゴに注目．

写真7　AT＆T社のブース．AT＆T社とRetics社は5月にOSIのインプリメンテーションに関しての発表を行なったばかりだが，ブースでは早くも「CP-1(OSI Communication Platform Release1)」のデモを行なっていた．

NeXT社自身は，会場での展示は行なわず，プライベートルームのみでのデモ．同社のRobert Poorによれば「NeXTはいわゆるUNIX屋ではないから，今回のUSENIXにはあまり力を入れていない」とのこと．一般のUNIXベンダーとは一味違った戦略を取っている．**写真 6** はNextStep上で動いているInfomix社のWingzだ．そのほかには，ボイスメールおよびDSPを利用した音楽関係のソフトを中心にデモをしていた．同社は，今年後半に予定されている68040ボードおよびカラーバージョンの発表に期待が集まっている．

今後のUNIXの行方は？

USENIXができて今年で15周年．これだけの歴史を誇るコンピュータOSのユーザー会はほかにはほとんど例がない．それはUNIXの歴史の長さを示すと同時に，UNIXが非常に使いづらいシステムであったことも証明している(UNIXが開発者とユーザー双方にとって完璧なシステムであったならば，こんなユーザー会など存在していなかったかもしれない)．

しかし，当初最もUser Unfriendlyだといわれたシステムも1つの節目にきていることは間違いない．X Windowなどの環境が整備されるとともに，その操作性も飛躍的に向上してきており，急速にビジネス分野にも浸透しつつあるからだ．マシンや環境にかかるコストもここ数年で急激に低下している．UNIXマシンの一部はすでに一般ユーザーの手に届く範囲にきている．今後の10年間もこのシステムが生き残ることもほとんど確実視されているが，単にサバイバルするだけでなく，同時にどのように拡散していくのか，その変化が注目されるところだ．

(ザイロンコーポレーション代表　脇山 弘敏)

写真 8　AT＆T社のブースにあった「Amiga3000」．System V　Release 4が走っていた．

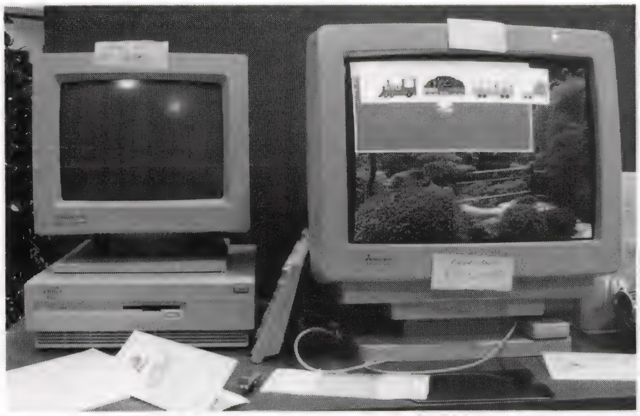
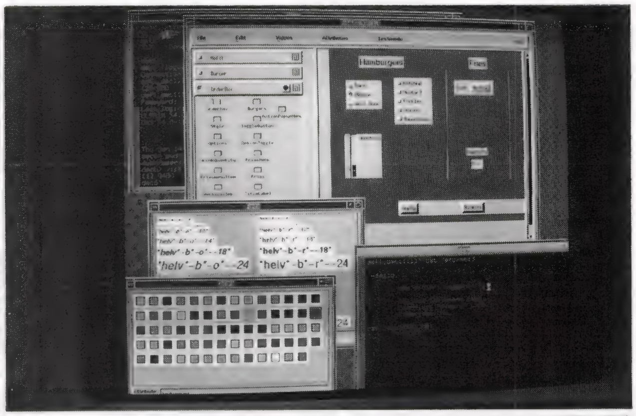


写真 9　DEC社のOSF／1のスナップショット．一応動いていたが，まだまだバグが多いという．



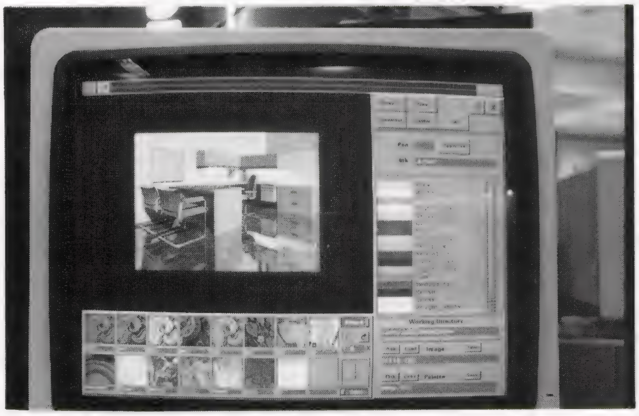


写真10　Intergraph社の「Intergraph5000」を使ったデモの1画面．

写真11　MT XINUのブースにて．一番右がUNIXのguruの一人Ed Gould．その左がマーケティングマネージャのAlan Tobey．

特別寄稿

USENIX Technical Conference Report

電気通信大学助手　砂原秀樹

日本には，「日本UNIXユーザー会」というUNIXのユーザーグループがありますが，アメリカには，「USENIX」と「UniForum」という2つの組織があります．UniForumは，UNIXでビジネスを行なっているメーカーやソフトハウスなどを中心とした団体で，商業色の強い組織です．これに対し，USENIXは大学や企業などで研究や開発を行なっている人たちを中心とした，アカデミック色の強い組織になっています．

USENIXでは，毎年1月と6月の2回，テクニカルコンファレンスを開いて技術者間の情報交換を行なっています．今回は6月11日から15日までの5日間，ロサンゼルスのダウンタウンから車で約1時間，アナハイムにあるアナハイムコンベンションセンターとアナハイムマリオットホテルの2会場で行なわれました．今年は，USENIXの設立15周年に当たり，記念のTシャツを参加者に配っていました．

コンファレンスは，2日間のチュートリアルと3日間のテクニカルセッションという構成になっています．また，展示会が併設されていて，各メーカーのワークステーションや書籍の展示が行なわれています．展示会のほうの話は別稿にゆずるとして，ここではテクニカルセッションを中心にお話しすることにします．

チュートリアル

最初の2日間は，チュートリアルです．チュートリアルというのは，大学の講義のようなもので，1つのトピックについて1日かけて詳しく解説を行なってくれるのです．今回のチュートリアルは表のようなものがありました．

僕が参加したのは，M2とT2というネットワークの話題でした．ともに，「UNIX Network Programming」の著者Richard Stevens氏で，1日目のM2は，TCP／IPプロトコルの概念，原理などの解説でした．なんとなく，Douglas Comer氏の「Internetworking with TCP／IP」（最近，共立出版から「TCP／IPによるネットワーク構築」というタイトルで翻訳が出されました）の講義のような感じでした．2日目のT2は，UNIX上でのネットワークプログラミングに関する話題で，まさしく「UNIX Network Programming」の講義でした．ともに，非常によくまとまっており，分かりやすく，僕には最適のチュートリアルでした．

ほかのチュートリアルには出ていないのでよく分かりませんが，面白そうだったチュートリアルはT4で，GNU ProjectのRichard Stallman自身がGNU Cコンパイラの内部構造について解説するものでした．次の機会には参加してみようかと考えています．

テクニカルセッション

6月13日からの3日間は，テクニカルセッションで11セッション34件の論文が発表されました．いつもと同様に，まずはプログラム委員長のJohn Mashey氏（MIPS社副社長）の開会の宣言で始まり，続いて基調講演です．

基調講演は，あのDennis Ritchie氏（UNIX生みの親の1人．もう1人はKen Thompsonですよ．Cの本の著者としても有名）が，「What Happens When Your Kid Turns 21？」という話をしました．彼は，USENIXには珍しくスーツにタイといういでたちで（USENIXはハッカーの集団なので，参加者も発表者もたいて

表　チュートリアルの内容一覧

6月11日
M1: Writing Applications with the OPEN LOOK Xt-Based Programming
M2: Introduction to the TCP/IP Protocol Suite
M3: Effective PostScript Programming
M4: Special Topics in C Programming
M5: Designing and Maintaining a LAN
M6: Programming with OSF/Motif
M7: UNIX on Modern Architectures
M8: Introduction to C++
M9: UNIX System V, Rel.4.0 Internals I, File, VM, and Process Subsystems
M10: Introduction to Programming the X Window System, Version 11
M11: Introduction to 4.3BSD Internals
M12: Mach Overview
6月12日
T1: XView, A User Interface Toolkit for the X Window System
T2: Advanced UNIX Network Programming
T3: Open System Interconnection (OSI) Principles
T4: Introduction to the Internals of the GNU C Compiler
T5: Topics in System Administration
T6: Introduction to Object-Oriented Programming
T7: UNIX Device Driver System
T8: Using C++ Effectively
T9: UNIX System V, Rel.4.0, Internals, Session & Streams Subsystems
T10: X Toolkit Intrinsics
T11: Beyond 4.3BSD: Advanced Kernel Topics
T12: Mach Virtual Memory Internals

いTシャツにGパンなのです）登場し，電話機を持ち出して電話でのインタビュー形式で話を進めていきました．

電話を取っては質問を受け，それに答えるといった感じで，UNIXとCの現状，Cの標準化，OSFとUI，C++などについて彼の意見を述べていきました．Bell研究所でのUNIXの最新版10th Editionまで飛び出し，なかなか面白いものでした．

最も面白かったのは最後のビデオで，RitchieとRob Pikeらが，Bell研究所での上司でありノーベル賞学者でもあるArno Penziasをだますというものでした．Ritchieたちが研究成果として「計算機が音声を認識し，それに対して応対を行なうシステム」ができたとしてPenziasにデモして見せるわけですが，種を明かせば，音声認識の結果というのは隣の部屋に設置してある端末にRitchieが打ち込んだものが表示されるだけの仕組み．応対するのは別室にいる人間で，ビデオカメラに向かって身振りをしていただけという内容でした．ともかく，それらしく見せるためにわざと認識に失敗してみたりして，会場は爆笑の渦でした．

基調講演のタイトル「What Happens……」は，Ritchie氏の子供のようなものであるUNIXが1969年に誕生して今年で21年目になるということに基づいています．

論文発表

論文はすべてを紹介することはできませんので，面白いと思ったものをいくつかピックアップすることにしましょう．

"Binary Emulation of Unix using the V Kernel"
by David R. Cheriton, Gregory R. Whitehead, Edward W. Sznyter — Stanford University

スタンフォード大学で開発中の分散オペレーティングシステムV Kernel（SUNはStanford University Networkの略だといわれるくらいで，Sunに載せられた最初のオペレーティングシステムはV Kernelだったのです）上で，UNIXのプログラムを実行する方法の話です．

"Unix as an Application Program"
by David Golub, Randall Dean, Alessandro Forin, Richard Rasid — Carnegie Mellon University

こちらは，カーネギメロン大学で開発中の分散オペレーティングシステムMach上でのUNIX機能の実現に関する話です．次のバージョンのMachでは，UNIX機能は1プロセス（UNIX Server）として実現され，カーネルの外で実現されます．この論文では，そのメカニズムについて述べられています．さらにUNIXだけではなく，MacOSやMS-DOSの機能を実現するサーバも開発中であり，発表の最後にビデオでは，Macintosh上でMachを動かし，その上でMacOSのエミュレーションを行なう（Macのゲームを実行させていた）というデモを見せてくれました．

"Evolving the Vnode Interface"
by David S.H. Rosenthal — Sun Microsystems

現在開発中の，SunOSでの最新のVnodeに関する話題です．Vnodeというのは，BSDのファイルシステムやNFSファイルシステム，CD-ROMのファイルシステムなど，異なる構造を持つファイルシステムを統一的に取り扱えるようにするための構造です．最新版のVnodeでは，Vnodeを積み上げることができ，複数のファイルシステムをまとめて特別な機能を持つファイルシステムを構築できるようになっています．たとえば，ローカルファイルシステムとNFSのファイルシステムをまとめると，ファイルキャッシュを構築できます（NFSでファイルの読み出しを行なうときにローカルディスクへその内容をコピーしておき，2度目の読み出しはローカルディスクで行なう）．

そのほか，バージョン制御やファイルシステムの多重化などを簡単に実現することができます．

"A Pageable Memory Based Filesystem"
by Marshall K. McKusick, Michael J. Karels, Keith Bostic — University of California at Berkeley

UNIXでのRAMディスクの論文です．RAMディスクの問題点として，容量が主記憶の容量に制限されることと，RAMディスクに利用した領域は主記憶として利用できないことなどがあります．これらの問題を解決するために，RAMディスクを実アドレス空間にではなく，ページングが可能な仮想記憶領域で実現するとどう

写真1　論文発表する東工大の太田昌孝氏．

なるだろうかという話が述べられました.

"A Fast /tmp File System by Delay Mount Option"
by Masataka Ohta — Tokyo Institute of Technology
Hiroshi Tezuka — Sony Computer Science Lab. Inc.

今回唯一の日本からの発表です. 発表者は, JUNETでは有名な東工大の太田昌孝氏です. 前述の発表の直後に「私はRAMディスクが嫌いです」といって話を始めたので, 非常に受けていました.

UNIXでは, ファイルを書き込むときにいったんメモリ上のバッファキャッシュに書き込んでからディスクへの書き込みを行ないますが, 普通はディスクへの書き込みを行なわないようなファイルシステム(Delayed File System)を用意して, それを/tmpなどの一時記憶用のファイルシステムとして用いるという話です.

"A Filesystem For Software Development"
by David Hendricks — Sun Microsystems

SunOS 4.1で導入されたTranslucent File Service (TFS) に関する話です. あるディレクトリにもう1つのディレクトリを重ねることで, バージョン制御などを行なえるようにしています.

たとえば, srcというディレクトリの下にソースコード群が格納されているとしましょう. このうちfunc.cというファイルを書き換えて新しい機能を追加する場合を考えてみます. 直接srcの下のファイルを書き換えるのではなく, newsrcというディレクトリの下にfunc.cをコピーして書き換えると, 古いバージョンを保存したまま新しいバージョンのファイルも管理することができます. ただし, これではばらばらになってしまうので, ディレクトリsrcとnewsrcを重ねると, func.c以外は古いバージョンのソースコードが見えるという状態を実現することができるようになります.

これら以外にもたくさん面白い論文がありましたが, 今回は誌面の都合でこの辺にしておきたいと思います. ともかく, 今回はテクニカルセッションの内容が濃く, 3日間ともわくわくするような内容が詰まっていました.

Bird of Feather

USENIXでもう1つ面白いのが「Bird of Feather (BOF)」といって, 特定のテーマに興味がある人たちが集まって議論などを行なうミーティングです. 今回も, GNU Project, UUNET/USENET, 4 BSD, Sprite, Project Athena, Multimax Usersなど, たくさんのBOFが行なわれ, たくさんの人が集まっていました.

僕は, GNU ProjectとSpriteのBOFに出ましたが, なかでもGNU ProjectのBOFでは, 例のLook and Feel問題について白熱の議論が繰り広げられていました. 僕は参加しませんでしたが, 4 BSDのBOFでは4 BSDの新しいバージョンの配布を6月末か7月初めに始めるというアナウンスがあったようです (これは, すでに4.3BSD-Reno Versionとして配布が開始されています). これには, OSIネットワーク機能やNFSなどの機能が追加されています.

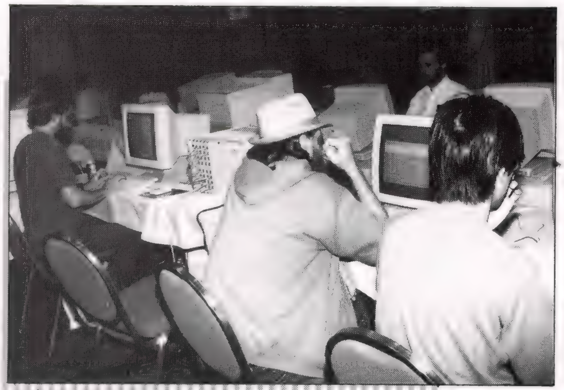


写真2　会場に併設されたターミナルルーム. X端末20数台が並んでいる.

その他

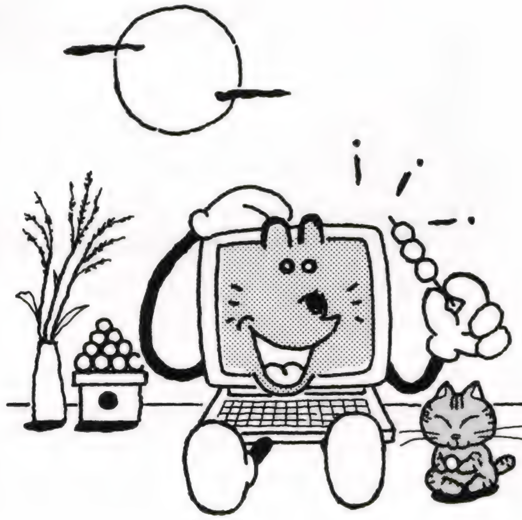
USENIXではいろいろな施設が用意されていて, その中で最も人気があるのがターミナルルームです. 今回は, 20数台のX端末を用意し, T1 (1.5Mbps) の回線を用いてNFSnetと接続していました. したがって, 会場にいながらにして自分の計算機にログインし, メールを読むことができるわけです. しかも, これが深夜の2時まで開いているのです.

最初は, 漢字フォントがなかったため, 日本語のメールが読めなかったのですが, 慶応大学の村井さんを通じてターミナルルームの管理をしているRick Adams氏にお願いして漢字フォントを置いてもらいました. その結果, 漢字のメールも読めるようになったわけです.

今回から新しいサービスとして, コンファレンスの参加者宛てにメールを送ることができるようになりました. これは, 「first_lastname@conference.usenix.org」宛てにメールを送ると, それがターミナルルームでプリントアウトされて, メッセージボードに張り出されるという仕組みになっています (残念ながら漢字のメールはプリントできません). 僕が, このメールアドレスを学生に教えておいたところ, 「ディズニーランドは面白かったですか?」とか「おみやげにこんなものが欲しい」といったメールが届きました.

USENIXのコンファレンスは, UNIXに興味を持つ技術者にとっては非常に面白いところです. 皆さんも参加されてはいかがでしょうか. 次回は, 1991年1月ダラスで開催されます.

イベント・カレンダー



8.16 → 8.31

1990 8 AUG.

16	木	
17	金	
18	土	
19	日	
20	月	Workshop on Programming Language Implementation and Logic Programming (Linkoping, Sweden)
21	火	Writing Better Computer Software Documentation (Atlanta, Ga.,U.S.A.)
22	水	コンピュータビジョン '90（筑波大学・茨城）
23	木	
24	金	
25	土	
26	日	
27	月	Compute USA '90 (Rio de Janeiro, Brazil) INTERACT 90 (Cambridge, U.K.)
28	火	1990年電気・情報関連学会連合大会（東京理科大学・東京） CAMSE '90（池袋サンシャインシティ・東京）
29	水	CAI学会第15回全国大会（工学院大学・東京） DEXA '90 (Vienna, Austria)
30	木	Sun アプリケーション・フェア 1990（新宿NSビル・東京）
31	金	

国内イベント

8月

○は開催地，◇は問い合わせ先を示す．

28-30
1990年電気・情報関連学会連合大会
●東京理科大学・東京
◆電子情報通信学会
(03)433-6691

28-31
CAMSE '90 コンピュータの材料科学・工学への応用国際会議・展示会
●池袋サンシャインシティ・東京
◆日刊工業新聞社
(03)222-7162

29, 30
CAI学会第15回全国大会
●工学院大学・東京
◆CAI学会(社)日本能率協会内
(03)434-0806

30, 31
Sun アプリケーション・フェア 1990
●新宿NSビル・東京
◆日本サン・マイクロシステムズ
(03)221-7228

9月

4-6
情報処理学会第41回全国大会
●東北大学教養部・仙台
◆情報処理学会 (03)505-0505

8-9
日本シミュレーション＆ゲーミング学会第２回大会
●籠谷大学理工学部・大津
◆(03)288-2576

10月

1-4
1990年度電子情報通信学会秋季全国大会
●広島工業大学・広島
◆電子情報通信学会 (03)433-6691

1-5
InfoJapan 90: International Conference on Information Technology
●京王プラザホテル・東京
◆情報処理学会 (03)505-0505

2-4
ソフトウェアショウ '90
●池袋サンシャインシティ・東京
◆ソフトウェア情報センター
(03)437-3394

2-7
第29回エレクトロニクスショー '90
●国際見本市会場・東京
◆日本エレクトロニクスショー協会
(03)284-1051

8-10
1st International Workshop on Algorithmic Learning Theory
●山上会館・東京
◆東京大学 大須賀 (03)481-4483
九州大学 有川 (092)641-1101

9-11
マルチメディア国際会議 '90
●幕張メッセ・千葉
◆日本経済新聞社 (03)544-7367

22-25
データショウ '90
●国際見本市会場・東京
◆日本電子工業振興協会(03)433-4547

23, 24
マイコンシステム＆ツールフェア '90
●幕張メッセ・千葉
◆(社)日本システムハウス協会
(03)567-5005

9.1 → 9.30

1990 9 SEP.

日	曜	イベント	日	曜	イベント
1	土		17	月	ICCD '90: International Conference on Computer Design (Cambridge, Mass., U.S.A.)
2	日	第１回国際文具・紙製品・事務機器展（幕張メッセ・千葉）	18	火	EP90: Electronic Publishing '90 (Washington, D.C., U.S.A.) 第３回アジア・パシフィックマイクロ波会議参加米国展示会（池袋）
3	月	EUROGRAPHICS '90 (Montreux, Switzerland)	19	水	インテリジェント アーバンシステム '90（国際見本市会場・東京） TELEMATICA: 国際テレコミュニケーション見本市 (Stuttgart)
4	火	情報処理学会第41回全国大会（東北大学教養部・仙台）	20	木	
5	水	データベース '90 TOKYO（池袋サンシャインシティ・東京）	21	金	サービスマーケティング '90（インテックス大阪・大阪）
6	木		22	土	
7	金		23	日	
8	土	日本シミュレーション＆ゲーミング学会 第２回大会（龍谷大学・大津）	24	月	SIGCOMM 90 (Philadelphia, Pen., U.S.A.) BROADBAND '90 FOC/LAN (Baltimore, Md., U.S.A.)
9	日		25	火	Fifth Annual PC EXPO in Chicago (Chicago, Ill., U.S.A.) Cenit Asia '90 (Hong Kong)
10	月	NetWorld 90 Dallas (Dallas, Tex.,U.S.A.) '90第１回マイクロメカトロニクス国際会議と展示会 (Berlin)	26	水	Potable Computing & Communications '90 (N.Y., U.S.A.) ネットワーキング ジャパン（流通センター・東京）
11	火	'90住宅設備展／ホーム・アメニティ '90（国際見本市会場・東京）	27	木	
12	水	Performance 90: 14th IFIP WG 7.3 International Symposium on Computer Performance (Edinburgh, Scotland)	28	金	
13	木		29	土	
14	金	印刷文化記念 '90 大阪印刷総合機材展（インテックス大阪・大阪）	30	日	
15	土				
16	日	Fourth Digital Signal Processing Workshop (N.Y., U.S.A.) KOFAPS '90 (Soul, Korea)			

海外イベント

8月

●は開催地，◆は問い合わせ先を示す．

15-17

WINDOWS／FALL '90

●Boston, Mass., U.S.A.
World Trade Center
◆CM Ventures, Inc.
5720 Hollis Street Emeryville,
CA 94608 (415) 601-5000

21-24

Writing Better Computer Software Documentation

●Atlanta, Ga., U.S.A.
◆Education Extension,
Georgia Inst. of Tech.,
Atlanta, GA 30332-0385;
(404) 894-2547

9月

3-7

EUROGRAPHICS '90

●Montreux, Switzerland
◆EUROGRAPHICS'90
Conference Secratariat Paleo Arts et Spectacles, P.O.Box 177
CH 1260 NYON Switzerland;
(41) 22-62-13-33

10-13

NetWorld 90 Dallas

●Dallas, Tex., U.S.A.
Dallas Convention Center
◆H.A.Bruno, Inc.,
385 Sylvan Avenue Englewood Cliffs, New Jersey 07632;
(201) 569-8542

18-20

EP90: Electronic Publishing '90

●Washington, D.C., U.S.A.
◆Dr. Lawrence A.Welsch,
EP90, National Inst. of Standards and Technology,
TECH B266, Gaithersburg, MD 20899; (301) 975-3345

25-27

Fifth Annual PC EXPO in Chicago

●Chicago, Ill., U.S.A.
◆H.A.Bruno, Inc.,
(201) 569-8542

10月

3-5

UNIX Solutions Exhibition and Conference

●Anaheim, Calif., U.S.A.
Anaheim Convention Center
◆ザ・インターフェイス・グループ・インク日本支社 (03) 237-8759

3-6

Seybold Computer Publishing Conference and Exposition

●San Jose, Calif., U.S.A.
San Jose Convention Center
◆Seybold Staff (213) 457-5850

4-8

COMDEX／Europe

●Paris, France. Paris Nord Villepinte Fairground
◆ザ・インターフェイスグループ・インク日本支社 (03) 237-8959

17-22

第21回韓国エレクトロニクスショー

●ソウル，韓国
◆韓国エレショー日本事務局
(03) 340-4698

10.1 ——— 10.30

1990 10 OCT.

日	曜	イベント
1	月	1990年度電子情報通信学会秋季全国大会（広島工業大学・広島） InfoJapan 90（京王プラザホテル・東京）
2	火	ソフトウェアショウ'90（池袋サンシャインシティ・東京） 第29回エレクトロニクスショー'90（国際見本市会場・東京）
3	水	Seybold Computer Publishing Conference and Exposition (San Jose, Calif., U.S.A.)
4	木	COMDEX/Europe (Paris, France)
5	金	
6	土	TAIPEI ELEC '90 台北エレクトロニクスショー（秋期） （台北，台湾）
7	日	
8	月	1st International Workshop on Algorithmic Learning Theory （東京 山上会館）
9	火	マルチメディア国際会議'90（幕張メッセ・千葉）
10	水	全日本オーディオ・フェア（池袋サンシャインシティ・東京）
11	木	
12	金	プリンティングフェア'90東京（幕張メッセ・千葉） ハイテク九州'90（福岡国際センター・福岡）
13	土	
14	日	
15	月	
16	火	
17	水	第21回韓国エレクトロニクスショー（ソウル，韓国）
18	木	第36回KOBEオフィスフェア（サンボーホール・神戸）
19	金	
20	土	
21	日	INTELEC '90 (Orlando, Fla., U.S.A.)
22	月	データショウ'90（国際見本市会場・東京） 国際コンピュータ統合化専門見本市・会議（Munchen, Germany）
23	火	マイコンシステム＆ツールフェア'90（幕張メッセ・千葉） INTER COMM '90 (Vancouver, B.C., Canada)
24	水	第6回ヒューマン・インターフェース・シンポジウム （総評会館・東京）
25	木	第27回情報科学技術研究集会（大阪国際交流センター・大阪） NORTHEAST COMPUTER SHOW (Boston, Mass., U.S.A.)
26	金	
27	土	
28	日	
29	月	
30	火	第4回バーコード及びIDシステム展（流通センター・東京） '90中部エレクトロニクスショー（中小企業振興会館・名古屋）
31	水	UNICOM '90 (Washington, D.C., U.S.A.)

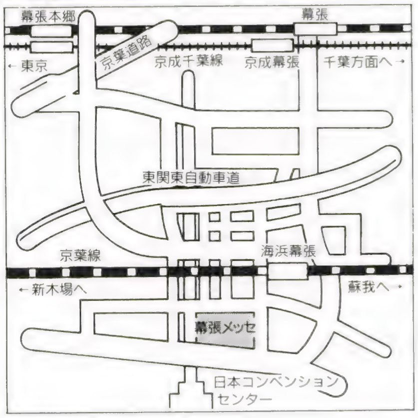
22-26
国際コンピュータ統合化専門見本市・会議
●Munchen, Germany.
Munchen Messegelande
◆在日ドイツ商工会議所見本市部
(03)593-1641

23-26
INTER COMM '90: Global Telecommunications Congress & Exhibition
●Vancouver, B.C., Canada
◆INTER COMM Office
P.O.Box 1616, Blaine, WA 98230

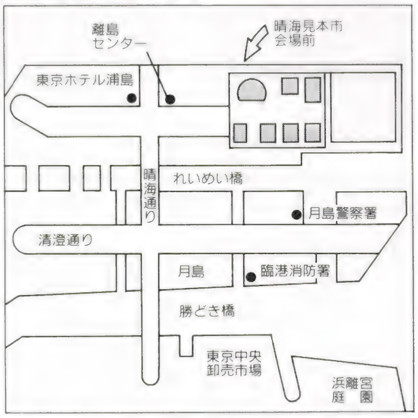
25-27
NORTHEAST COMPUTER SHOW
●Boston, Mass., U.S.A.
Bayside Expo Center
◆ザ・インターフェイス・グループ・インク日本支社 (03)237-8759

31-11/2
UNICOM '90
●Washington, D.C., U.S.A.
◆Susan Ryba (202)296-9800

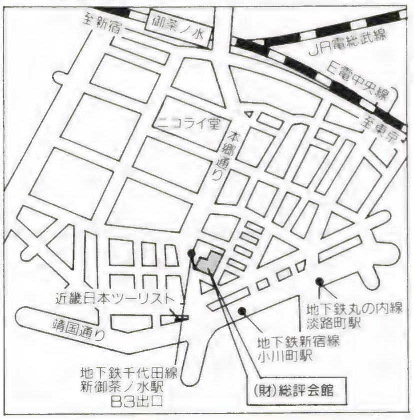
□幕張メッセ



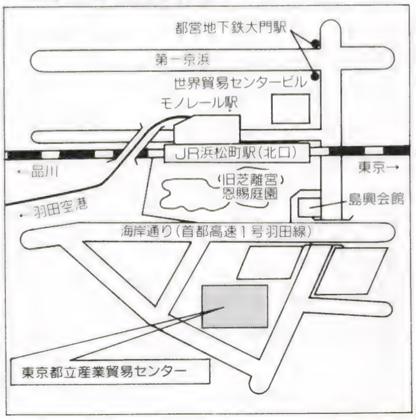
□国際見本市会場



□総評会館



□東京都立産業貿易センター



特集

オペレーティングシステムとは何か…

オペレーティングシステムの基本であるタスク管理，メモリ管理，ファイル管理のそれぞれについてMS-DOSの限界を明らかにし，マルチタスクOSの機能がいかに必要であるかを述べる．また，そのOSの機能の実現に使われている技術について簡単に紹介する．さらに，今後のOSで一般的になるであろう新しい機能についても解説していく．

OSとMS-DOSはどこが違うか

CG/Meta Corporation

常駐プログラムの限界

MS-DOSとMacOS

パソコンのシステムをフレキシブルに使おうとすると，たった1つのプログラムにすべてを任せるわけにはいかない．しかし，パソコンのアプリケーションに自分だけの機能を追加して使おうと思うと，それなりの工夫をしてやらなければならない．

常駐プログラムのプロセス切り替え

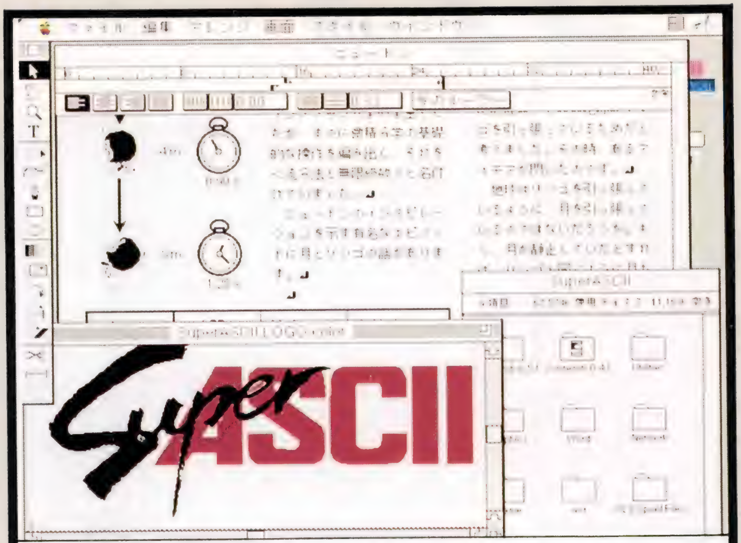
常駐プログラムは，MS-DOSの世界に欠かせないものになっている．プリントスプーラのprint.exeや，PDSのスクリーンセーバなどのツール群は，すべてこの系統のプログラムだ．これらのプログラムは「常駐」という名のとおり，メモリに常駐して処理の必要が発生したときだけ行動をとるようになっている．カナ漢字変換プロセッサなども，ドライバという少し違った組み込まれ方のプログラムではあるが，同じような動作のしかたをするプログラムだ．

この種のプログラムは，タイマやキーボードなどからのさまざまな割り込みをいったん受け取り，割り込みの種類やいろいろな条件をもとに処理の必要があるかどうか調べる．時計を表示するプログラムならタイマ割り込みを監視していればよいし，カナ漢字変換のようなプログラムならキーボードを監視していればよい．普段は何もしないでMS-DOSに制御を戻す．MS-DOSのほうは，ほかの割り込み処理プログラムに制御を渡したり，割り込みが発生する前に動いていたプログラムに制御を戻すことになる．時計の分針を動かすべき時刻になったり，特別なキーシーケンスやホットキーが押されたりして処理の必要が生じた場合は，常駐プログラム本体が必要な処理をした上で，何事もなかったかのようにMS-DOSに制御を戻すことになる．

常駐プログラムは本当に必要な機能の部分のほかに割り込み監視の部分が必要である分，通常のプログラムより複雑になっていることが分かると思う．また，割り込み処理を低いレベルで行なうことになるから，かなり難しいプログラムを書くことになる．また割り込み処理をする部分は，常駐プログラム本来の処理が必要になる可能性があるかぎり，メモリに常駐していなければならない．

ここで，このようなプログラムのための機構が，あらかじめシステムに用意されているMacintosh(以下Mac)の場合を考えてみよう．Macには，デスクアクセサリというジャンルのツールがあって，どのプログラムを使っているときでも自由にメニューから呼び出せる．電卓やメモ帳などがいつでも使えるのは，非常に便利なものだ．

このデスクアクセサリは，デバイスド



Macintoshでは，デスクアクセサリとマルチファインダという機構を使って，いろいろなプログラムをいろいろな形で同時に使うことができる．データのやり取りが簡単にできるし，ちょっとした機能がいつでも使える．

Inside Operating System

割り込み処理ルーチン

割り込み処理の流れは，ハードウェアによって実行中のプログラムが中断されるところから始まる．この時点で，CPU内部の状態はすべてメモリに待避される．ここで，あらかじめ割り込みの種類に応じて設定された割り込み処理ルーチンが実行され，このルーチンが終了時にもとのプログラムの処理を続行させる．割り込みは，実行できない機械命令を実行しようとしたとき，I/Oが終了したとき，ソフトウェアによって割り込みを起こしたとき，タイマやI/Oなどからのイベント，リセットスイッチが押されたり，電源が入ったときなどに起きるが，CPUのアーキテクチャによって種類は異なる．通常，コンピュータは割り込みによってハードウェアの状態を受動的に監視するが，ポーリングによって能動的に監視する場合もある．

ライバの1種として実現されている．処理はオープン，クローズ，コントロールという3種類のメッセージに対応して進められるようになっている．コントロールのメッセージでは，メニュー選択などのイベントが送られてくる．また，時計など間欠的に処理をする必要のあるプログラムでは，定期的に送られてくるメッセージを利用すればよい．もとのアプリケーションが終了したときに，メモリ領域の解放など特別な後始末の処理をする必要のあるデスクアクセサリのためにも，専用のメッセージが用意されている．

これらのメッセージに対応して処理を進めるだけでよいのだから，MS-DOSの常駐プログラムの場合のように自分で割り込み監視をしなくてもすむ．基本的なプロセス切り替えの機構が用意されているといってよい．このような機構については，あとでゆっくりと説明しよう．

常駐プログラムのシステムコール

もう1つの重要な違いは，デスクアクセサリが呼び出されたときの状態である．MS-DOSのホットキーで起動される常駐プログラムなら，起動された瞬間，ほかのプログラムが何をやっているか知りようがない．ホットキーが押された回数をカウントするために自分のメモリ領域中の変数を書き換えるという程度のことは他のプログラムに迷惑をかける心配なくできるが，それだけでは外に対しては何もできない．画面に何か表示するにせよファイルに何か書き込むにせよ，意味があることをしようと思えば，システムコールを呼んだりBIOSをコールしないわけにはいかない．

ところがこれらのシステムで提供されている機能は，いつでも安全に使えるわけではない．ほかのアプリケーションが，自分のプログラムの中だけですむような処理をしていてくれればよいのだが，ちょうどシステムの機能を使おうとしているところだと危険がある．たとえば，アプリケーションが通常のMS-DOSのシステムコールを使って，ファイルに書き込みにいっている最中になんらかの原因で割り込みがかけられたとしよう．この割り込みの結果，常駐プログラムが呼び出され，さらにこのプログラムが同じシステムコールを使ってファイルへの書き込みをすると問題が起きる．たとえ別のファイルに対する書き込みであっても，システムコールが使っているバッファの中身を書き換えてしまうことになる．したがって，このままもとのアプリケーションに制御を戻しても，もとの書き込み処理はうまく終わらない．

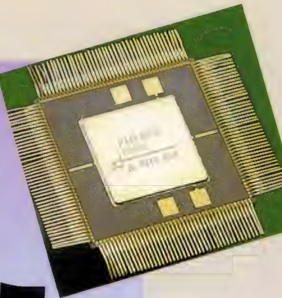
しかし，このような事態を回避する方法はある．通常の常駐プログラムでは，ディスクへの書き込みをする場合は，通常のシステムコールを使わない．代わりに自分でI／Oバッファやバッファの管理ルーチンを用意し，もっと低レベルのI／Oルーチンを呼び出す．低レベルのI／Oルーチンはもともと割り込み禁止状態で動いているので，このレベルではもとのプログラムとの干渉が起きることはない．結果として両方のプログラムとも無事にI／Oができるわけだ．

Macではこの問題を，もとのアプリケーション側が自分で安全だと判断した場合だけ，明示的にほかのプログラムを実行するようにして乗り切っている．プログラムの書き方についてしっかりした規約のあるMacでは，これでよい．しかし，オペレーティングシステムのほうで解決できないわけでもない．いつ要求が入ってきても順番に処理できるようにしておけばよいのである．しかしこの機構の実現のためには，実はオペレーティングシステムにとって非常に本質的な排他制御の問題が起きてくる．このことについてもあとのページで説明しよう．

キー入力は本来のアプリケーションに処理される前に，常駐プログラムの割り込み処理ルーチンを通る．ここで，特定のキーの組み合わせが発見されたら，アプリケーションに渡さず，常駐プログラムの本体が起動される．この本体が書き込み処理をし，もとのアプリケーションの書き込み処理とたまたま衝突したりすると，問題になる．

常駐プログラムの限界

割り込みによるタスク切り替え



個別のプログラムが特に意識せず，ほかのプログラムと共存できる環境を作るためには，割り込みによるタスクの切り替えが通常必要になる．このような環境を作るには，ディスパッチャというタスクの切り替えをするプログラムが重要な働きをする．

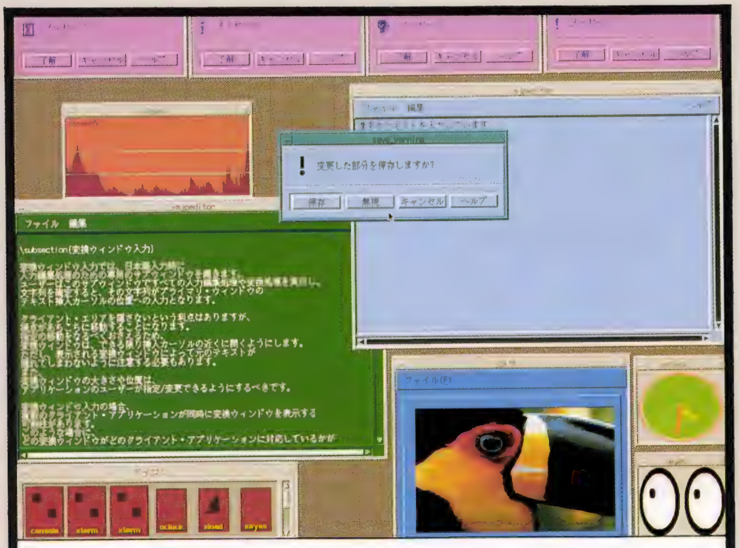
井戸掘りのタスク切り替え

並列処理の難しさを語るときに，よく井戸掘りの話が引き合いに出される．1本の井戸を掘るのに1人でやって10日かかったとしたら，10人では何日間かかるかという問題である．10本の井戸を掘るのなら1人でやる場合の1/10の時間で片が付くかもしれないが，1本の井戸に同時にかかれる人数は限られている．10人でかかっても10倍の速度で仕事が進んだりはしない．特別な手法を用意して初めて井戸掘りが速くできるようになる．この話の井戸はなすべき仕事，つまりメモリやCPUを使って走っているプログラム（タスク），CPUが井戸を掘っている人ということになる．

このたとえに従うと，通常のマルチタスクの環境は井戸がたくさんあって人が1人しかいないようなものだ．本当の井戸なら，1本掘り終えてからほかの井戸に手をつければよいようなものだが，タスクの場合はそうはいかない．ユーザーの入力待ちやディスクのアクセス待ちなど，さまざまな理由によって井戸掘りが進まなくなることがある．その隙にほかの井戸を掘り進めなくてはいけない．

すでに紹介したMacのタスク切り替えでは，井戸を掘っている途中で井戸のほうから仕事を切り替えてくれという要求が出される．その結果システムという別の井戸に作業が移り，そこでの指示によりさらに別のタスクへの切り替えが発生することになる．掘るたびに別の場所の地図が出てくる宝探しのようなものだ．井戸を掘っている側であるCPUは，無理なく仕事の対象を移していくことができるが，井戸であるタスクのほうには細工が必要である．タスクのありようを規定しているプログラムに，あらかじめ適当なところでシステムに制御を移せという命令を書いておくことになる．

MS-DOSの常駐プログラムの場合は，まれにしか用事の発生しない特別な井戸が，他の井戸を掘っている人を強制的に一時呼び付けて，突然自分の井戸の仕事をさせる．その仕事が終わり，特に用がなくなったらもとの井戸に戻らせる．ある場所での仕事中に，よそで突発的な事故が起きた場合のような進め方になる．普通の井戸，すなわちプログラムのほうは，大した手間がかからないが，常駐プログラムには面倒が付きまとう．



ワークステーションでは，多くのプログラムが同時に機能できるようになっているのが普通である．このような能力によって，プログラムのエディットをしながら，動作を確認してデバッグするなどという使い方ができる．

Inside Operating System

ディスパッチャの最適化

ディスパッチャは，何があろうとも1秒間に何回かは起動してもらわないといけないタスクから，頻繁に動くべきであるが少しずつしか処理の必要のないタスク，トータルの処理時間のみが問題で多少処理の間があいても問題のないタスクといったふうにタスクを分類し，どのタスクを処理するかを決めるようにしなければいけない．このためにタスクごとのCPU使用量や，ユーザーの設定などをもとに優先度を決定する．

ユーザーの要求などに対する反応時間をターンアラウンドタイム，一定時間内に処理できる仕事の量をスループットといい，システムの性能の基本的な2つの尺度である．この矛盾する要求をうまく満たすのが，ディスパッチャに与えられた使命であり，設計の難しいところである．

ディスパッチャのタスク切り替え

現在一般的な高機能のオペレーティングシステム，たとえばUNIXやOS/2，さらにIBMの大型汎用コンピュータのオペレーティングシステムであるMVSなどは，マルチタスク機能を持っている．これは，多くの井戸を1人で効率良く並行して掘り進めるためのシステムになぞらえることができる．

これらのシステムでは，基本的にMS-DOSの常駐プログラムの機能を拡張した機構でマルチタスクが実現されている．とはいえ，すべてのプログラムが割り込みを監視して，必要に応じて制御を奪い取るといったことではない．そのようなやり方では，すべてのプログラムが複雑な割り込み処理ルーチンを抱えて割り込み処理を行なわねばならなくなる．割り込みが発生するたびにこれらのルーチンが順次起動されるのだから，システム全体として割り込み処理に取られる時間が長くなり，どのプログラムも本来の処理を進められなくなる．また，割り込み処理中にほかの割り込みが起こることが多くなる．その結果，割り込み処理ルーチンが割り込みを禁止して動いていれば次の割り込みを要求される時間内に処理できなくなる．割り込み可能にしていれば次々に割り込みが続き，どの割り込み処理も終わらなくなる．

そこで先に挙げたシステムでは，割り込み処理を1つにまとめて，適当な割り込みを受けた時点でどのタスクを起動するか決定するディスパッチャあるいはスケジューラと呼ばれるルーチンを起動し，そのルーチンがほかのすべてのプログラムを起動する（制御を移す）ようにしている．ほかのタスクは，あらかじめこのディスパッチャに，どのようなときに起動してほしいか登録しておくことになる．

これが標準的なマルチタスクのオペレーティングシステムの基本的な機構である．先ほどからの例でいうと，作業の指図をする現場監督のようなものが，どこかの井戸の中にいて，呼び付けては指示を出しているということになる．

作業をしてもらっている井戸，つまりそれぞれのタスクは，通常はファイルを読みにいった時点やユーザーの入力待ちになった時点で現場監督，つまりディスパッチャに制御を戻すことになる．ファイルなどのアクセスやユーザーの入力などの面倒はオペレーティングシステムが見てくれることになっているから，制御をディスパッチャに戻すのは簡単である．

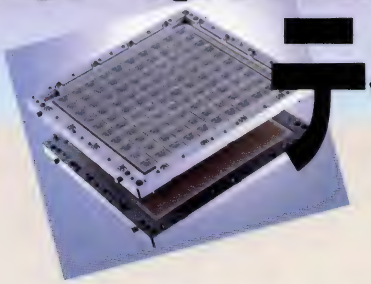
また，ずっとCPUを使い続けているタスクの場合にも，ディスパッチャに制御を戻す道は用意されている．一定時間ごと，通常は数十ミリ秒に1回，タイマ割り込みという割り込みをハードウェアによって発生させ，そのたびにディスパッチャがどのプログラムを起動するか決める．また，ほかのプログラムが待っているファイルの読み込みや，ユーザーの入力が発生したときにも，ディスパッチャが起動される．このような機能があるから，無限ループに陥ったプログラムがあっても，ほかのタスクは処理を続けられることになる．

逆に，ディスパッチャがこのような機能を実現するためには，勝手なところで処理を中断されたタスクの処理を，あとで好きなときに再開できるようになっていなければならない．これはオペレーティングシステムの割り込み処理ルーチンの責任であって，割り込みをかけられたタスクの環境をタスクごとに保存し，ディスパッチできるようにしておかなくてはいけない．具体的には，CPUのレジスタやスタック，ヒープなどの状態を保存しておかなくてはいけない．それだけでなく，タスク側もいつ処理を中断されてもよいようにしていなければいけない．

ディスクへの入出力の終了による割り込み，タイマからの割り込み，キー入力による割り込みなどによって，今まで動いていたタスクが中断され，ディスパッチャが起動される．ディスパッチャは，あらかじめ登録されているタスクの性格，タスクごとの今までのCPU使用の度合いなどを参考にして，どのタスクを起動するか決定する．

常駐プログラムの限界

衝突防止のためのテクニック



多くのタスクが同時に動くようになると，1つのタスクしか動いていなかったときのように，システムの資源をすべて無条件に独占して使うことはできない．必要なときに必要な資源だけ独占するように配慮して，プログラムを書かなくてはならない．

信号による衝突防止

交差点には信号がある．普通は一定時間ごとに赤青が切り替わって，どちらか一方の道路からの通行しかさせないようになっている．しかし，これはたしかに衝突防止には有効であるが，あまり効率的な管理とはいえない．たとえ横から車が来なくても，信号が赤なら意味もなく信号が変わるのを待たなくてはいけない．そこで，車や人が来たときだけ1方向に切り替えるようになっている押しボタン信号を考える．これなら，無駄に信号待ちをさせられることはない．

マルチタスクのコンピュータには，このような切り替えの機構が必要になる．多くのタスクが同時に1つのファイルに書き込みをしようとすると，なんの交通整理の機構もなしでは，いつか衝突が起きてファイルが壊れてしまう．1つのプログラムがひととおり書き込みを終えて安全な状態に戻るまで，ほかのプログラムはちゃんと待っていなければいけない．

ファイルなど交差点に相当するpublicな領域に書き込みをするときには，処理を始める前に信号を赤にして，ほかのプログラムが処理に入れないようにする．処理が終わったら信号を青にして，ほかのプログラムが処理を開始できるようにする．多くのタスクに，publicな資源をうまく順番に使わせるためのcriticalな処理の切り替えが，排他制御というオペレーティングシステムの基本的機構である．この機構はファイルへの書き込みだけでなく，デバイスの制御，ディスパッチャの実現など，オペレーティングシステムの機能のほとんどに関係している．

しかし排他制御は，それほど簡単には実現できない．押しボタン信号の場合で，押しボタンを押すと一瞬で信号を切り替えられるようになっていたとしよう．ほぼ同じタイミングで，2方向からの車が押しボタンを押せば，当然衝突が起きる．ある車がボタンを押してから交差点を通過し終わるまでは，ほかの方向からボタンを押しても信号が変わらないようにロックしておかなくてはいけない．

オペレーティングシステムの場合にも，さまざまなプログラムが押しボタンを押しては，criticalな処理をすることになる．もっとも単純に押しボタン信号を実現するには，処理に入る前に割り込み禁止にして，処理が終わったら割り込み可

大型汎用コンピュータでは，ファイルやタスクをコントロールするためのデータ構造体である，各種のコントロールブロックの排他制御が非常に重要になる．不用意なプログラムは，特定のタイミングでのシステムダウンをまねく．

Inside Operating System

モニタの構成

ディスクなどのI／Oをする場合，通常かなり長い時間がかかるため，実際にI／Oを開始したあと終了するまでの間にも，ほかのタスクがそのデバイスへの要求を並行して出せるようにするための手続きとデータ構造のペアが必要になる．このI／Oなどの処理を始めてから終わるまでの間の面倒を見る機構を，モニタと呼ぶ．モニタは，そのタスクに関するデータをモニタ内に保持したまま，さまざまな原因に応じてwaitさせ，さらにほかのタスクの処理を開始させなくてはいけない．モニタ内でwaitさせてあるタスクが，ハードウェアなどの都合により，動けるようになった場合は，ただちにそのタスクを起動し，モニタから出ていくようにする．モニタは，入り口と中の待ち条件のそれぞれについてセマフォを用意すれば実現できる．

能にすることになる．MS-DOSのレベルではこうやるしかないわけだが，これではその処理とはまったく関係がない処理を緊急にやろうとするプログラムも，足留めされることになってしまう．多少工夫してやると，どこか特定の変数を信号として使って，それが赤になっているときは，時々信号を見ながら待つということになる．しかし，これではCPUが信号待ちをしているだけのタスクのために使われることになってもったいない．

セマフォによる排他制御

マルチタスクのオペレーティングシステムの排他制御には，このような信号による管理の方法を発展させたセマフォというやり方が使われることが多い．セマフォという信号機に，タスクから“wait”と“signal”という２つの操作の要求を出すことによって制御をする．ただし，この信号機はあらかじめどれだけの数のタスクが同時に通行できるか決められるようになっている．セマフォはこの数を初期値として記憶していて，通行しているタスクの数によって増減させる．この数字そのものが信号の表示でもあって，さしあたりは正の数字が青信号，０が赤信号と考えてよい．交差点のように一度に１つしか通行できない場合は，初期値は１になり，誰かが通行しているときは０になるわけだ．－1になるときは誰かが信号を無視していて，衝突の危険があるということになる．余談になるが，現実には数字の増減もタスクに任されているので，衝突が起きるような場合は実は数字の操作も行なわれない．律義に信号を操作してから衝突しにいくようなプログラムは誰も書かないからである．

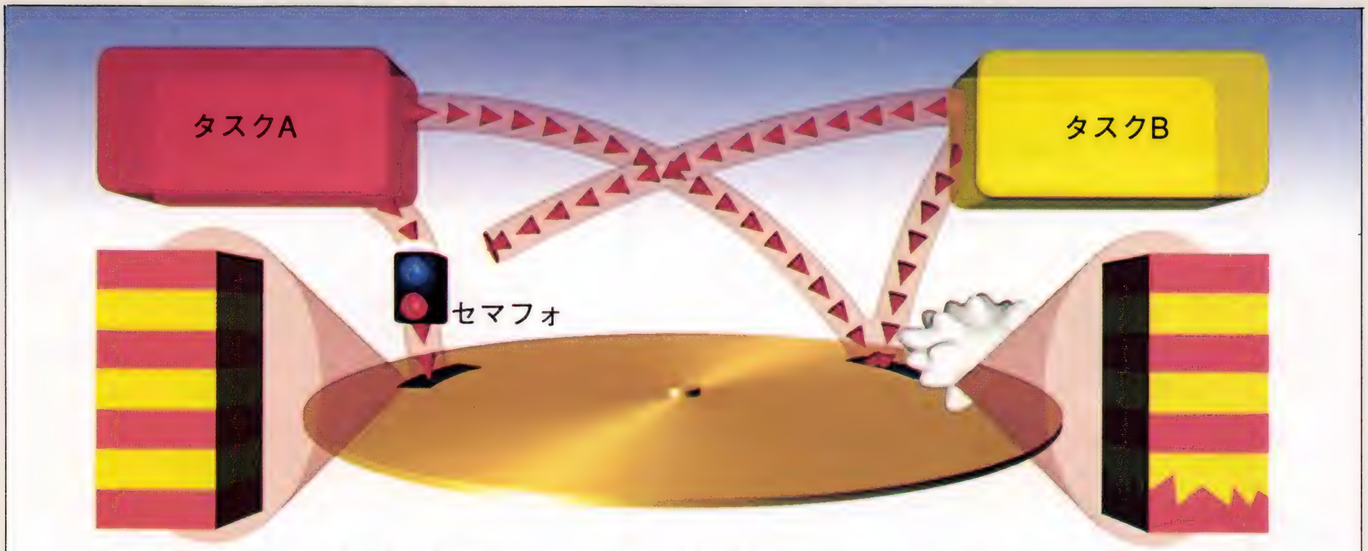
waitは交差点あるいは通路を通過しようとする操作であって，セマフォの値を見て正ならセマフォの値を1減らして処理を続ける．これでセマフォの値が0になると，ほかのタスクから見ると赤信号ということになる．その場合は本当にそのセマフォに対してwaitする．

signalは交差点を通過し終わったことの通知である．もしそのセマフォについて信号待ちをしているタスクがあれば，信号が変わったことを伝える．この結果少なくとも1つのタスクが新たに動けるようになるから，ディスパッチャを呼ぶことになる．誰も待っているタスクがなければ，セマフォの値を1増やす．なおセマフォの値が正の場合は，当然ながら誰も待っていないはずである．

オペレーティングシステムでは，通常はハードウェアや各種のバッファなどをあるタスクに専有させるためにこのような機構を使う．同じバッファを複数用意しているような場合には，セマフォの初期値を用意してあるバッファの数にすればよい．また，ディスパッチャ自体もこのようなセマフォの機構と密着して作られている．

しかし，このセマフォの実現には，さらに細かいレベルの排他制御が必要になる．ボタン信号の例でいうとボタンを2人以上が同時に押した場合に相当する事態が起きる可能性があるのだ．セマフォの値を調べてから変更するまでの間に，ほかのタスクが値を変更して処理を開始したりすると，衝突が起きる．当然だが，この衝突を回避するためにはセマフォを使うことはできない．

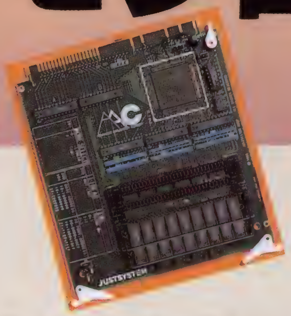
CPUが1つのシステムでは，この間は割り込み禁止にしてほかのタスクが動かないようにする．ところがCPUを複数持っているシステムの場合は，割り込みを禁止してもほかのタスクが処理をしないという保証がない．そこで1命令で値を調べると同時に変更する，特別な命令を用意して排他制御する．このように細かな単位の排他制御をもとに，大きな単位の排他制御ができるのである．



２つのタスクが，たまたま同時に同じファイルにデータを書き込みにいくとファイルが壊れてしまう．実際に同時ではなくても，操作が２フェイズに分かれるようなものでは，その途中にほかの操作が起きれば同じことである．これを防ぐには，書き始めてから書き終わるまで，ほかの書き込みを禁止するようにロックをすればよい．

フラグメンテーションする640Kbytes

メモリ管理の危険



パソコンのメモリは，すべてアプリケーションに引き渡され，そのアプリケーションの内部だけで管理されることが多い．しかしマルチタスクのシステムでは，メモリのダイナミックな取り扱いのために，タスク間の調整をするシステムが必要になる．

細切れにしないメモリ管理

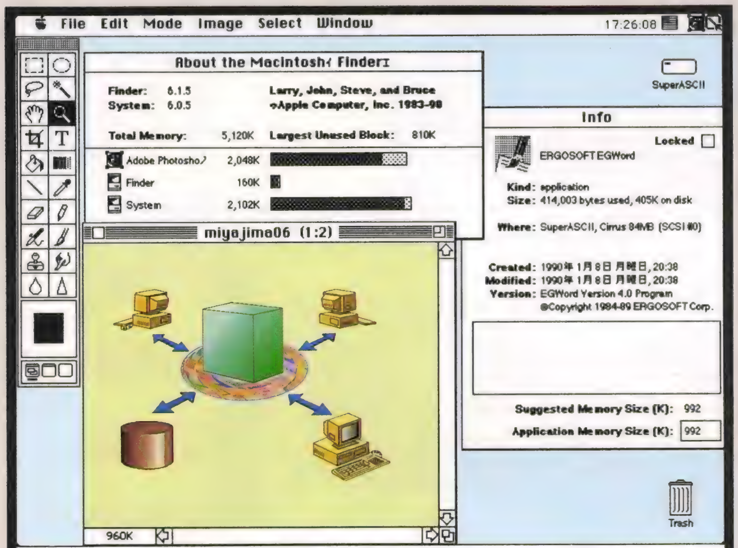
MS-DOSが，そのメモリに関して持つ問題のうち最大のものは，量の上限である．640Kbytesでできることには限界がある．しかし，MS-DOSでは一時に動くタスクは基本的に1つしかない．タスクは細かいことを気にせず，ポインタが及ぶかぎりのメモリを使うことができる．CPUが使い放題なのと同じことである．

しかしMS-DOS環境でも子タスクの起動をしようとすると，メモリの管理をする必要がでてくる．子タスクの起動のための空きメモリを作るには，さしあたり不必要なメモリを解放してやらなければならない．子タスクに最大限度までメモリを使わせるには，一時的にしか使わないメモリは必要がなくなりしだい解放しておき，処理の必要に応じてまた獲得することになる．これによって，同じメモリをいろいろな用途に使うことができ，固定的に使う場合よりメモリの使用効率が良くなる．

しかし一般に，このようなダイナミックなメモリ管理をすると，新たに難しい問題が起きてくる．ある時点にAというメモリ領域を獲得したとしよう．そのあとBというメモリ領域を獲得し，さらにAを解放したとする．この時点で使用中のメモリ領域はBだけである．しかし，ここで子タスクを起動しようとしても，最初からBだけを獲得したときより小さいタスクしか起動できない．最初は連続して空いていたメモリ領域の途中にBという領域が取られているため，2つに分かれてしまったのだ．このようにメモリが細切れ(fragmentation)になってしまうと，子タスクの起動などで連続した大きなメモリ領域を獲得しようとしたときに問題が起きる．

Macの場合は，ダイナミックなメモリの獲得と解放を非常に頻繁に行なうように設計されているので，メモリが細切れにならないようにするための対策が用意されている．解決策は単純で，使用中のメモリ領域を適当なタイミングで中身ごと動かし，端のほうにまとめてしまうのである．

具体的には，プログラムがメモリ領域を要求したとき，オペレーティングシステムは状況に応じてメモリ領域のデータを移動してしまう．このためにMacでは，メモリ領域はゾーンという大きな単位で管理し，その中に個別のメモリ領域



Macintoshのマルチファインダ環境では，アプリケーションごとのメモリ消費をモニタすることができる．アプリケーションごとのメモリの最大使用量のデータは，それぞれの製作者の手によってリソースとして設定されている．

Inside Operating System

名前のバインディング

プログラム中に表われるシンボル名を，アドレス空間上の論理アドレスに変換するのは，内部名についてはコンパイラやアセンブラ，外部名についてはリンケージエディタ，あるいはローダということになっている．しかし，いくつかのオペレーティングシステムでは，外部名からアドレスへの変換が，オペレーティングシステム自身によって行なわれる．このようなシステムでは，実行時にダイナミックにライブラリの参照が行なわれ，必要なライブラリがすでにメモリ上にロードされていれば，それが共用される．

このように可能なかぎり遅い時期までメモリを名前によって管理すれば，フレキシブルにメモリを使えるようになる．しかし，ハードウェアの支援なしでは，パフォーマンスの低下の可能性がある．

を取り，それらのメモリ領域へのポインタを保存するようになっている．メモリ領域を移動するときには，排他制御をしたうえでメモリ領域をゾーン内で移動し，ゾーン内にあるポインタを変更する．このシステムで用意しているポインタを使うかぎりは，メモリを使う側のプログラムはメモリ領域の移動について特に意識する必要がない．

しかしプログラム内のワーク変数として，そのメモリ領域の途中を指しているポインタを使っている場合などまでは，システムは面倒をみてくれない．したがって，そのようなポインタはメモリ獲得などメモリ領域移動の原因となる操作をしないところで，局所的に使うようにするとか，ゾーン内のポインタからのオフセットを使うようにするとかといった必要が生じる．そのような配慮ができない場合は，メモリ領域の移動を一時的にあるいは永久的に禁止することもできるが，そのような移動禁止は当然メモリが細切れになる危険を伴う．

Macではプログラムが置かれているメモリ領域についても，ダイナミックな管理がセグメント単位でできるようになっている．このようなダイナミックなメモリ管理は，オペレーティングシステムの基本的な機能の1つである．

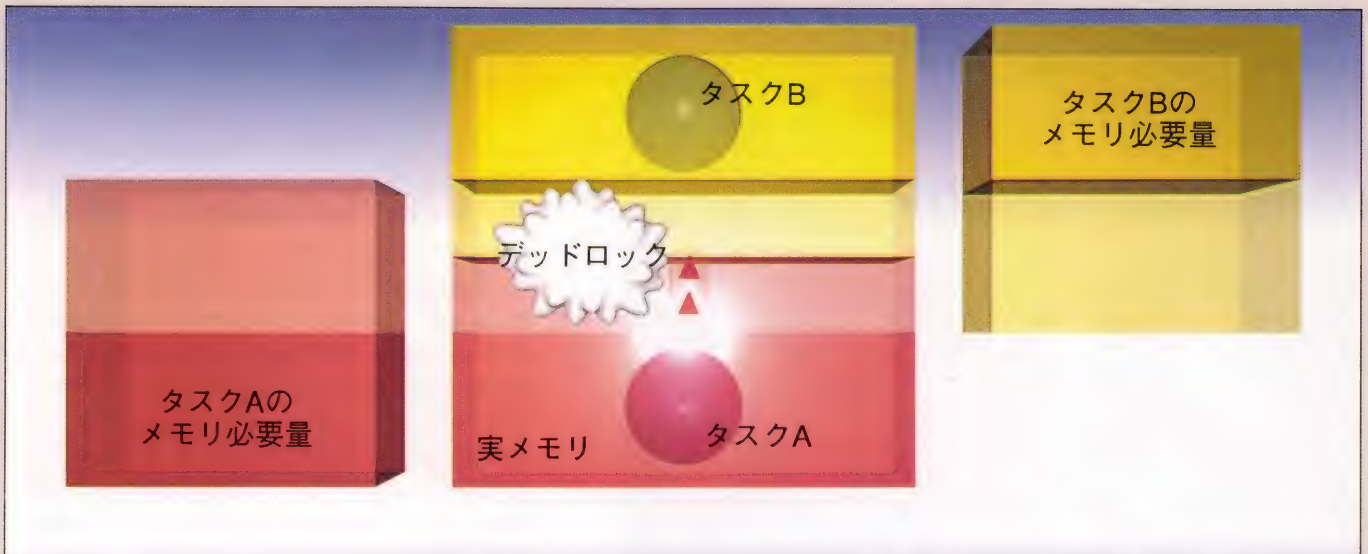
メモリ保護と特権モード

マルチタスクのオペレーティングシステムでは，メモリについてもう少し別の種類の問題が起きるようになる．動いているプログラムが増えると，どれか1つが暴走する可能性は高くなる．その結果システム全体が暴走したのでは，誰かがプログラムの開発をしている間はシステム全体が使いものにならなくなる．このような事態を避けるために，通常のプログラムは，ほかのプログラムの領域に書き込みができないようになっている．

このような機構は通常メモリ保護と呼ばれ，ハードウェアで実現されている．メモリをあらかじめブロックに分けておき，タスクごとに使うべきブロックの番号を決め，読み出し，書き込み，プログラムの実行のどれを許すかを指定しておく．こうすれば，同時に走っているほかのプログラムにデータを破壊されるようなことはなくなる．また複数のタスクでデータを共有することもできる．

しかし，メモリ保護の機構を活かすには，あるタスクが自分に与えられた以外のブロックをアクセスできる特権モードも必要である．ファイルの読み書きなど，オペレーティングシステムが持っているルーチンを呼び出さないかぎりできはしない．かといって，いつでも簡単にそのようなモードになれたのでは，システム全体の暴走を防ぐという目的が果たせない．モードの変更には特別な機構が必要になる．このため通常は，ソフトウェア割り込みによって特権モードに変わるようにしている．実は割り込みが起きた瞬間から，制御はオペレーティングシステムに渡っているのである．逆にオペレーティングシステムが普通のタスクに制御を戻すときも，割り込み発生前の状態に戻す特別な命令が使われる．通常の割り込み処理と同じように前の処理が継続される．

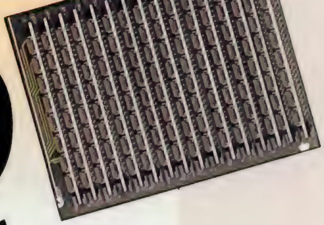
もう1つ，マルチタスク環境下では，メモリの獲得に制限をつけるべき場合がある．これは本来タスク管理の問題なのだが，メモリのダイナミックな獲得をそれぞれのタスクに任せると，どのタスクも終了するまでに必要になるメモリを取れなくなる場合が起きてくる．このような事態を回避するためには，あらかじめそれぞれのタスクが最大限必要とするメモリ量をオペレーティングシステムに登録しておき，少なくとも1つのタスクは終了してメモリを解放することになる．



それぞれ固有の上限の枠内で，ダイナミックにメモリを取るタスクが2つある．これら2つが無事に処理を完了するためには，少なくとも一方の上限まではメモリが取れるだけの空きメモリが常に残されていなければいけない．中途半端にメモリを取らせると，どちらの処理も終わらないデッドロックに陥る可能性がある．

フラグメンテーションする640Kbytes

仮想記憶の原理と機能



メモリを好きなだけ自由に使えればプログラムを書くのは非常に楽になる．プログラム上特に意識せず，実際に存在するメモリよりはるかに広大なメモリ空間を使うための機構が，仮想記憶である．現在はページング方式によるものが一般的である．

ページによる仮想記憶の実現

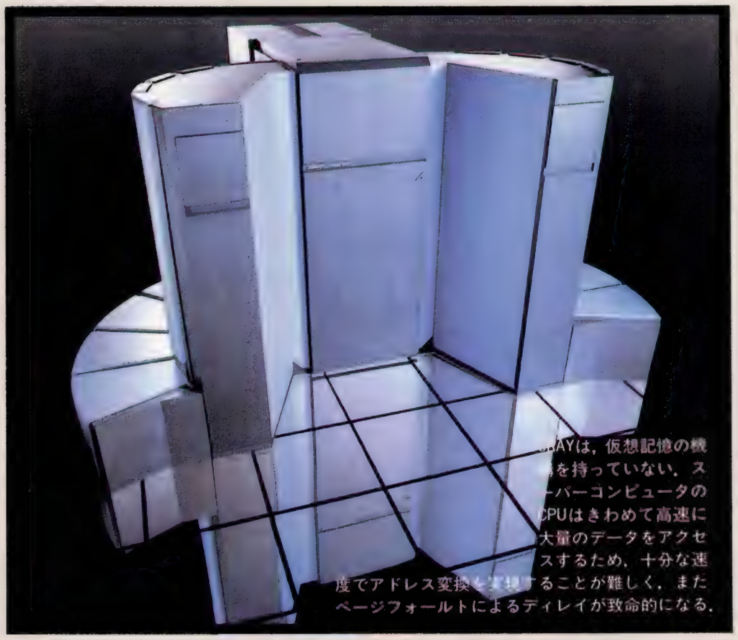
片手に1本しか持てない短剣を，手だけを使って3本持ち歩くのは，なかなか熟達が必要なことではあるが，不可能なことではない．現実にジャグラーという人々は，そんな芸を見せてくれる．短剣をほうり上げて，落ちてくる前にもう1本の短剣を投げ上げる．あとは受けとめるときにしくじらなければよいのである．

足りない手を増やすのにはこんな手法が使われているのだが，足りないメモリを増やすのにも，これと似た手法が使われている．この仮想記憶という手法では，必要になったメモリは，すべて存在することにしてしまう．32bitアドレスのハードウェアなら，4Gbytesのメモリから好きなように使って，この使われた分のメモリがすべて付いていることにしてしまうわけである．

そのうえで，この仮想メモリというものを一定の大きさ（4 Kbytesあたりが一般的）に刻んでしまう．この刻んだものをページといって，これを単位にしてジャグルするわけである．実際に存在するメモリを手とすると，さしあたりいらない分のページをほうり上げていさえすれば，仮想メモリのページならいくらでも持てるということになる．

で，メモリをほうり上げる先はどこかというとディスクである．といっても，本当にジャグラーのようにディスクとメモリを往復させるわけではない．どちらかというと，ディスクを倉庫代わりに利用するというイメージである．メモリの空きが足りなくなったら，さしあたりは使わなさそうなページを実メモリからディスクに追い出してむりやり空きを作る．逆に，本来なければならないはずのページがメモリ上になければ（ページフォールト），そのページをディスクから読み込んでくる．このようにすれば，実際にあるメモリより大きな仮想メモリを使うことができる．

この方式による仮想記憶の最大のメリットは，ページの操作が楽にできることである．不揃いな大きさの荷物を倉庫に詰めるより，同じ大きさに揃えられたコンテナを詰めるほうが，取り扱いは楽である．ジャグルの場合でも，大きさが違うものをいろいろ取り混ぜてほうり上げようとするとタイミングを計るのが難しくなる．ページの場合，もしサイズが不



CRAYは，仮想記憶の機構を持っていない．スーパーコンピュータのCPUはきわめて高速に大量のデータをアクセスするため，十分な速度でアドレス変換を実現することが難しく，またページフォールトによるディレイが致命的になる．

Inside Operating System

ページ待避のアルゴリズム

ページングのアルゴリズムの背景にあるのは，プログラムの局所参照性という事実である．プログラムは，そのフェーズごとに特定の範囲のアドレスを繰り返し参照する．このため，最も長く使われずにいるページが，次の瞬間に使われる可能性の最も低いページである．したがって実メモリが不足したときは，そのようなページからディスクに追い出せばよい．現実には，過去の一定時間内にアクセスされた回数の最も少ないものを追い出すアルゴリズムが使われることが多い．なお，このときに実メモリの必要を発生させたタスク自身からページを奪うローカル方式のアルゴリズムもあるが，現在はすべてのタスクについてメモリの使用状況を検討してページを奪う，ページスティーリングという方法が一般的である．

揃いならしまう場所（ページフレーム）の管理が大変だ．サイズが同じだから，実際のメモリ（実メモリ）の空きは残っているページの数だけで管理できる．またページの大きさは通常2のべきになっているから，実メモリ上のページフレームの位置もアドレスの上位何ビットかだけを管理すればよい．このアドレスの変換には仮想メモリのアドレスから引けるテーブルを用意すれば，検索による時間はかからない．

この結果，ページによる仮想記憶は，ハードウェアで比較的容易に実現できる．これは，非常に重要なことである．ページフォールトが起きたときはともかく，実メモリにページがあった場合はすべてハードウェアで処理できないと，仮想記憶は使い物にならない．プログラムがメモリをアクセスにいくたびに1ステップ余計に動いたとしたら，実行されるステップ数は数倍になってしまうのだから，このことは明らかだろう．

これらの機構をハードウェアで実現しても，ディスクを使う以上ページフォールトが起きればミリ秒単位のディレイが起きるのは当然である．このディレイの影響を少なくするページフォールトを減らすための管理機構も，ハードウェアで実現してやることになる．

仮想記憶の拡張とその限界

このようにして実現した仮想記憶にも，限界がないわけではない．アドレス空間の狭いコンピュータの場合，システム全体にわたって1つのアドレス空間を共有したのでは，仮想メモリ自体が足りなくなる．このためアドレス空間を，タスクごとに別々に仮想記憶で用意するという手法が使われる．これを多重仮想記憶といって，現在のオペレーティングシステムでは一般的になっている．タスク間の干渉を防ぐ意味でも有効な手段である．

しかし，このように仮想メモリ空間が広大になると，それによる問題も起きる．仮想メモリ上のアドレスから実メモリのアドレスやディスク上のページフレームへのアドレス変換には専用のアドレス変換テーブルを使うのだが，このテーブルが巨大になる．このテーブルは，仮想メモリ上のページの数だけエントリがあって，それぞれにディスク上や実メモリ上のアドレス，メモリへのアクセス権の管理のためのフラグなどが書かれている．また，このテーブルでは実メモリ上のページがディスクへ書き出す必要があるかどうかを確認するために，ディスクから読み込まれてから以後変更されたかどうかチェックするフラグなども持っている．

またアドレス空間が4Gbytesで，ページのサイズが512bytesであったとすると，仮想メモリのページの数だけエントリがあるのだから，8Mエントリ必要になる．こんな巨大なものを実メモリに置いておくわけにはいかない．このテーブルを仮想記憶に置くシステムも多いが，1ページの仮想メモリをアクセスするのに，何回もページフォールトが起きる可能性があり，性能上は問題がある．この結果，特にアドレス空間の巨大なシステムでは，アドレス変換テーブルの検索にハッシュを採用して，実メモリのページの数しかエントリがいらないようにする方式も使われている．

ページ方式のデメリットは，ソフトウェアの論理的構成とはまったく関係なくメモリが分割されていることである．タスク間でプロシージャを共用するといった目的やメモリアクセス管理には，任意の長さのセグメントを単位とするセグメント方式の仮想記憶のほうが，プロシージャ単位でディスクとのやり取りができて都合がよい．そのようなわけでページによる仮想記憶のシステムでも，セグメント方式が併用されていることが多い．

格子状に区分したメモリを，ハードウェア(実メモリ)の量に関係なく定義し，実際にタスクに使われた部分から，実メモリを格子の1マス(ページ)単位に割り当てていく．実メモリが存在する以上にメモリが必要になった場合は，実メモリに置かれているものの中から，比較的アクセスされていないものをページ単位でディスク上の専用の領域に追い出してやる．

EXE, COM, BAT, TXTしかいらない?

リソースによるファイル管理

きわめてプリミティブなレベルでは，ファイルは単にディスク上のブロックあるいはシリンダの集合体である．このレベルの上に，ソフトウェアによる抽象化を何階層かにわたって繰り返し，データを蓄えるための道具として使えるファイルにしていく．

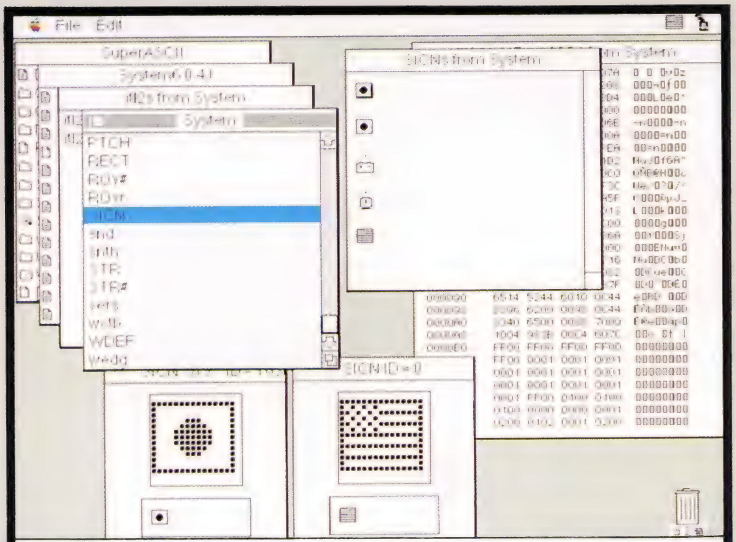
ソフトウェアによるファイルの実現

MS-DOSのファイルは，すでにかなり抽象的な存在である．ファイルは名前によって識別される文字データもしくはバイナリデータの列として取り扱われ，このバイト列の中の任意の場所から読み書きなどの操作ができるようになっている．“TYPE”を使えばテキストファイルを先頭から読みだすことができるし，“SYMDEB”を使えばバイナリファイルを任意の場所から読み書きすることができる．また，書き込みによってファイル自体の大きさを増減させることもできる．もちろんプログラムからこれらの操作をするのは格別難しいことではない．MS-DOSが提供している，INT21Hシステムコールの機能を利用するだけでよい．

この機能の実現のためにMS-DOSに与えられたハードウェアは，ディスク上に散らばった同じサイズのブロックである．これに対してはブロック番号を指定して読み書きできるだけである．これをファイルとして扱うためには，散らばったブロックを順番に並べてやる必要がある．MS-DOSはこのためにFAT（File Allocation Table）やディレクトリなどファイル管理のためのデータが書かれたブロックを参照し，そこに書かれた番号のブロックのデータをつないでいく．

さらにプリミティブなレベルでは，ディスク上のブロックを実現するのもオペレーティングシステムの機能であるが，近ごろではここまで物理的なレベルは，メディアの物理フォーマットの問題も含めて，専用のコントローラでサポートされていることが多い．こうして物理的な磁気ディスク上でのデータの表現やその読み書きのためのヘッドを移動する機構の制御というレベルにまで到達する．

ここでは，ある番号のブロックをアクセスするという行為は，以下のような物理的処理に分割される．まずヘッドをそのブロックを含むシリンダ(ディスクを輪切りにした同心円の1つ，何枚ものディスクが同軸上にある場合は，それぞれの面の上で中心軸から同じ距離にある円の組)まで移動する(シークと呼ぶ).次にシリンダが回転して目的のブロックが出てくるのを待つ(サーチと呼ぶ)．さらに目的のブロックの開始を，フォーマット時に決めてある特定のデータによって検知し，ディスクの回転に合わせてそのブロ



Macのリソースを，アプリケーションに合わせてインタラクティブに定義するResEditというツールの画面．アイコンの形をResEditに組み込まれたペイントツールで編集しているのが分かる．しかし，おなじみの16進ダンプに頼らなければいけない面も残っている．

Inside Operating System

入出力の同期と非同期

プリミティブなシステムでは，ファイルの入出力は当然タスクの処理の枠内で進められる．ディスクの読み込みを行なうのもタスク自身であるから，原則的にはその間にほかの処理が行なわれることはない．しかし，近ごろのハードウェアは，特定のブロックに対して読み込み要求を出せば，読み込みが終了した時点で割り込みをかけてくれるから，その間にほかの処理ができないわけではない．ディスパッチャは，このような入力待ちの間にほかのタスクを起動しにいく．通常のタスクのレベルでも，このように入出力待ちの間にほかの処理を行なうことができる．このように入出力が終了するまでほかの処理を進め，終了時の割り込みによって元の処理を進める方法を非同期の入出力と呼び，これに対して一般の入出力を同期の入出力と呼ぶ．

ックを読み込みあるいは書き込む．

このように物理的な機構から，ファイルという概念を実現するのは，オペレーティングシステムの基本的機能の1つである．

ファイルシステムとファイルの抽象化

しかし，このような抽象化には必ず付きまとう問題がある．物理的ファイルのレベルでは，今述べたように，それぞれの操作に時間がかかる．また，ブロックの数は有限であって，自由に数を増やしていけるわけではない．にもかかわらず，ユーザーから見えるファイルは，自由に大きさが変えられ，いくらでも作れ，高速でアクセスできなければならないわけである．

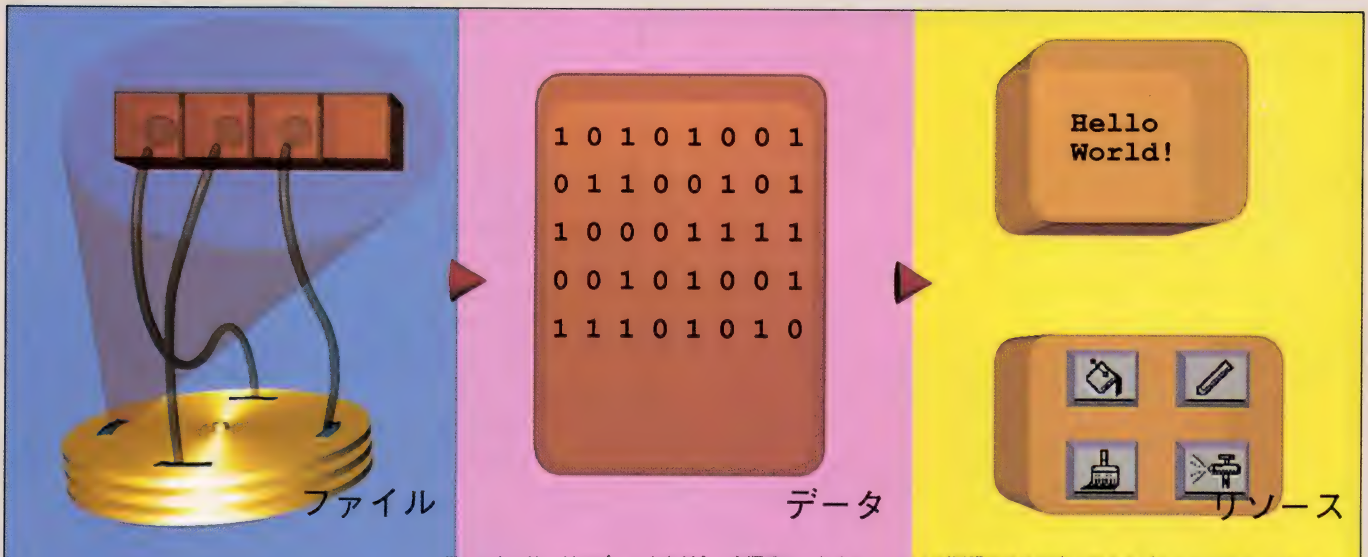
これらの条件のうちの1つだけを満たすファイルシステムを作るのは，それほど難しくない．高速化するためには，1ファイルを1シリンダに割り当てるようにすればよい．ディスクアクセスにかかる時間のほとんどがシークにかかる時間なのだから，ファイルの中でシークが発生しなければ高速になる．ところがこれでは，ファイルのサイズがシリンダのサイズを超えられない．この方針のまま，自由にファイルの大きさを変えられるようにするには，シークを最小限に押さえるため，連続したシリンダをつないで使うことになる．このためには，シリンダをファイルごとにあらかじめ最大の大きさまで確保しておかねばならない．

ファイルが物理的限界まで大きくなれるようにするには，MS-DOS の場合のようにディスクブロックをチェインしたものをファイルとすればよい．ところが，これではファイルの作成と削除を繰り返しているうちに，1つのファイルを構成するブロックがディスク上で物理的にばらばらの位置に散らばってしまう．これではファイルを単に頭から読むだけで，ブロックの数だけシークが発生し非常に遅いものになってしまう．

両方とも欠点だらけの方式のようだが，実は両方とも実際のオペレーティングシステムで使われている．前者は大型汎用コンピュータ，後者はパソコンや UNIX で使われている方式だ．ただし，どちらのシステムでも欠点を克服するための手法は使われている．UNIX では，ブロックを再使用するときにばらばらにならないよう，いったんソートするなどの最適化がなされている．大型コンピュータでは，VSAM（Virtual Storage Access Method）と呼ばれるデータベース的な機能を持つインターフェイスを用意することによって，フレキシブルなアクセスができるようにしている．

ここまでで，バイト列などのデータとしてのファイルは実現できている．一般には，バイナリデータにどんな意味を見いだすかは，ユーザーの責任である．しかし，Mac ではさらに高いレベル意味を持つデータまで，ファイルシステムが提供している．アイコンの形やメニューの構成，メッセージの文字列，画像などプログラムが標準的に使うデータを，アプリケーションのファイルにリソースという形で追加できる．これらのリソースは，GetNamedResource などのルーチンでメモリに読み込めば，そのまま表示用ルーチンなどに渡して使えるようになっている．プログラマが文字列や画像など論理的に高いレベルのデータを，実際のデータ表現に注意する必要なく，取り扱えるという点が重要である．

このようにファイルシステムは，ディスクやテープなどの物理デバイスから物理的なデータ，さらに論理的なデータへとアクセスの抽象化を進めるシステムである．しかし，その抽象化の過程で，アクセス効率が下がらないようにしなければならない．



ディスクには，どのブロックをどういう順番につなぐとファイルが再現できるか書いてあるブロックがある．これを見てデータを再現する．Mac の場合プログラムなど本来のデータに，プログラム実行上必要になるアイコンやメニュー，ダイアログなどの雑多なデータを1つのファイルにまとめて保存できるようになっている．

EXE, COM, BAT, TXTしかいらない?

ファイルの概念の拡張

ファイルシステムは，単なるファイルの集合体ではない．ファイルのアクセスを一元的に管理するための基盤であると同時に，ファイル名からファイルへの検索のためのデータベースでもある．当然このようなシステムは，さまざまな応用が可能だ．

ディレクトリが作る名前と管理の空間

ファイルシステムを考えるとき，名前の管理の問題を忘れてはならない．ファイルシステムの本来の姿は，名前をもとに，特定のデータが呼び出せるというものである．名前さえ指定すればいつでも同じデータが読めるということは，ファイルシステムの重要な特徴の1つである．

しかしMS-DOSにおいては，名前からデータへの対応は一意には定まらなくなっている．単にファイル名のみを指定した場合，どんなデータがアクセスできるかは，その瞬間ユーザーなりプログラムなりが，どのボリューム，またどのディレクトリにいるかによって決まる．

このように，状況に応じて名前の持つ意味が変わるのは，コンピュータが複数のユーザーあるいは複数のプロジェクトによってシェアされている場合などには有効である．フロッピーごとに違うアプリケーションが立ち上がるようにしているシステムなどが，その典型である．マルチユーザーのオペレーティングシステムであるUNIXでもMS-DOSと類似したディレクトリ構造が使われており，アプリケーションの初期設定ファイルを各自のホームディレクトリに置いておくというような使い方も一般的である．

しかし，このようにディレクトリ構造を持つファイルシステムにおいても，名前とデータとの対応を固定できないわけではない．MS-DOSでもボリューム名から始まる絶対パス名を指定すれば，どのディレクトリにいようと，必ず同じファイルを参照することができる．この結果，システムに存在するファイルの名前は単に辞書順に並べられるものではなく，2次元的な階層構造を持った複雑なものになる．どれだけファイルを作ろうと名前が足りなくなることはないし，ユーザーの使い勝手によってファイルのディレクトリを分けることもできる．

UNIXでも，同様に絶対パス名を指定することはできるわけだが，1つ重要な相違がある．MS-DOSの場合は物理的ボリュームがそのままファイル名の指定に登場してしまうが，UNIXの場合はそのシステム上のすべてのファイルが1つのディレクトリから参照できる．すべてのディレクトリが直接にせよ間接にせよ，“/”というルートディレクトリの下にあるのだ．ファイルが実際に置かれているデ

```
ドライブ A: のディスクのボリュームラベルは DOS_3'3B
ディレクトリは A:\EDITOR

.               <DIR>     90-02-05   20:23
..              <DIR>     90-02-05   20:23
VZ       COM    45970     90-01-10    1:51
VZ       DEF    12328     90-05-14   15:29
VZS      COM    38450     90-01-10    1:51
VZS      DEF    10294     89-12-10    1:50
VMAP     COM     1958     90-01-10    1:04
EZKEY    COM     2787     90-01-10    1:21
DOC             <DIR>     90-02-06   11:24
SRC             <DIR>     90-02-05   20:38
MIFES    EXE    98408     89-04-06    4:12
MIFES    HLP    42412     89-04-07    4:12
MIKEY    EXE    91400     89-03-30    4:12
MILOGO   COM     3015     88-09-03
MIKEY    DEF    16402     89-03-15
       15 個のファイルがあります.
  4610048 バイトが使用可能です.
```

MS-DOSのディレクトリ情報．日付やサイズだけが表示されている．このほかに「不可視」，「システム」などの属性フラグがある．

```
% ls -lF
total 21
drwxr-xr-x  4 wnn       512 Jul 25 23:34 dic/
drwxr-xr-x  3 root      512 Jul 25 23:34 doc/
-r-xr-xr-x  1 root     4667 Jul 25 23:38 hinsi.data*
-r--r--r--  1 root      705 Jul 25 23:38 jserverrc
-r--r--r--  1 root      705 Jul 25 23:38 jserverrc.sys
drwxr-xr-x  2 wnn       512 Jul 25 23:38 rk/
drwxr-xr-x  2 wnn       512 Jul 25 23:38 rk.vi/
-r--r--r--  1 root      996 Jul 25 23:38 wnnenvrc
-r--r--r--  1 root      846 Jul 25 23:38 wnnenvrc.omr
-r--r--r--  1 root      871 Jul 25 23:38 wnnenvrc.rem
-r--r--r--  1 root      996 Jul 25 23:38 wnnenvrc.rev
-r--r--r--  1 root      932 Jul 25 23:38 wnnenvrc.sys
-r--r--r--  1 root      996 Jul 25 23:38 wnnenvrc_R
-r--r--r--  1 root      846 Jul 25 23:38 wnnenvrc_R.omr
-r--r--r--  1 root      871 Jul 25 23:38 wnnenvrc_R.rem
-r--r--r--  1 root      996 Jul 25 23:38 wnnenvrc_R.rev
-r--r--r--  1 root      941 Jul 25 23:38 wnnenvrc_R.sys
```

UNIXのディレクトリ情報．アクセス権の範囲，リンクの数，所有者，所有者のグループ，サイズ，日付，名前が表示されている．

Inside Operating System

ネットワークへの拡張

ファイルシステムは，システムの枠を超えて広がっている．ルートディレクトリの前にユニークなマシンアドレスを付けて，ネットワーク上のほかのコンピュータのファイルを指定するシステムもある．UNIXではほかのコンピュータ上の物理的なファイルシステムを，自分のファイルシステムにマウントすることができるNFS（Network File System）も一般的になっている．

このようなシステムを作ること自体は技術的に完成されつつあるが，ネットワーク上のどこか仮想的な場所にトップレベルのディレクトリを作るのが難しいのと同様に，管理の一元化は難しい．ルートディレクトリや特権ユーザーの確定が難しいためである．このため，これらに代わる技術が研究されつつある．

ィスクが違っていて，物理的には独立したファイルシステムであっても，同じように参照する．ルートディレクトリを含むファイルシステムの適当なディレクトリの下に，ファイルシステム全体を置く(マウント) できる．

このようにディレクトリ構造がツリー構造になっているシステムでは，すべてのファイルやディレクトリが，より上位のディレクトリからポイントされているからこそアクセスできるのである．あるディレクトリの所有権を与えられれば，そこから下のすべてのファイルについて，基本的には所有権を持つことになる．もちろん，自分の持つディレクトリの1つについて，ほかのユーザーに読み書きやファイルの作成を許可することができるし，その結果ほかのユーザーが作った，すなわち所有するファイルが自分のディレクトリの下に存在することもある．それでも，ディレクトリそのものの所有権は元の持ち主にあるのだから，ファイルの削除だけはできる．

このようなファイルの階層構造を利用して，アクセス権やアカウントを管理することができる．ルートディレクトリに対応するルートという特権ユーザーがシステム全体の管理権を持ち，個別のユーザーやプロジェクトの管理者に対してディレクトリを分け与えるという構造になる．当然，各ユーザーは必要に応じてディレクトリを作り内部の細分化をすることになる．ファイルシステムの階層構造が，そのままアクセス権やユーザーの管理の階層構造としても使われているわけだ．MS-DOS のように個別のデバイス中のファイル階層だけではなく，システム全体のファイルの階層が設定されているからこそ，システムで一元化された管理が可能になったわけである．

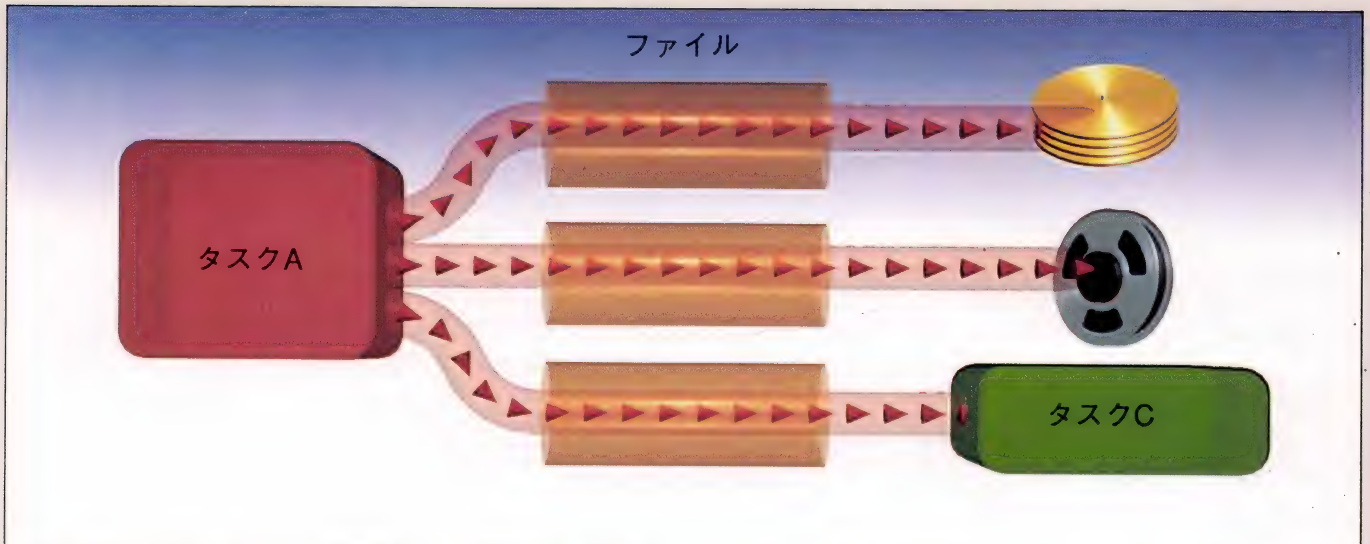
名前と管理のための空間を流用する

UNIX では，このようにして作られたファイルシステムの管理機構と名前の機構を，さまざまな目的に流用している．ディスクやテープドライブ，イーサネットインターフェイス，メモリ，シリアルポートなどを特殊なファイルとして取り扱い，物理的に読み書きできるようにしている．これらはディスクのダンプやリストア，カーネルでサポートされていないプロトコルのパケットの読み書きなどに使われる．

これらの特殊なファイルのアクセスには，通常のファイルと同じ open，close，read，write などのルーチンが使われるが，物理デバイスの性格上，可能なオペレーションには制限がある場合もある．またボーレートなどの特殊なパラメータについての設定が必要なシリアルポートなどの場合には，ioctl というルーチンを呼んでやる必要がある．

通常これらの特殊ファイルは“/dev”というディレクトリの下に root の所有するファイルとして置かれている．しかし，この位置は管理上設定されたものであって，名前と物理デバイスを実際に対応付けるのは，管理者の責任あるいは権利である．

同様の名前による管理が，同一コンピュータ内(UNIX ドメイン)のプロセス(UNIX ではタスクとほぼ同義に使われる)間通信にも使われている．あるプロセスが，ほかのプロセスにサービスするルーチンをサーバとして提供するときには，特殊なファイルを作る．サービスされる側(クライアント)は，あらかじめこの名前を知らされていて，この名前に対してサービスを要求してくる．ファイル自体の存在にはほとんど意味がなく，名前がシステム内でユニークに定まることに意味がある．すでにその名前のファイルがあれば，それ以上サーバは動けないわけである．



UNIX ではディスクやテープなどの入出力機器をダイレクトに使うときの指定や，ほかのプログラムとの通信チャンネルを指定するために特殊なファイルを使っている．このような特殊な機能の名前がユニークになることが保証されると同時に，通常のファイルのアクセスと同じプログラムインターフェイスが使えるというメリットがある．

新しいOSの機能

コンピュータが貴重であった時代には,1台のコンピュータを大勢で使うのが当然であった.現在のOSの基本はこの時代に作られたものであり,1人がたくさんのタスクを使う環境のための機能は,今も研究中である.

Inter Process Communication

1人のユーザーの1つの目的のために,プロセス(あるいはタスク)を同時に数多く使うには,プロセス相互間のサービス依頼や調整のための,プロセス間通信の機能が必要である.

プロセスは,1つのメモリ空間内で動く,ほかから独立したプログラムのことであった.古典的にはそのとおりで,ファイルのコピーのプログラムやコンパイラのように,ディスクとのやり取りしかしないようなものなら,オペレーティングシステムとだけ関連を持っていればすむ.またオペレーティングシステムは,そのようなアプリケーションに合わせて作られたわけである.

ところがパソコンのアプリケーションは,ファイルだけでなく通信ポートからのデータやユーザーのキーインなどさまざまなイベントに対応して動作しているものが多い.このようなプログラムを作るには,次にどれがくるか分からないいろいろな外因をすべて並行して監視しなければならない.

最も簡単にこのような機構を実現するには,各種のポートを順番にチェックして何も起きていないことを確認するようにすればよい.何か起きていたら,それに応じた処理をすればよいのである.しかし,この方法では,何も起きていないことを確認するだけのためにCPUが活動しなければならない.これは,マルチタスクOSの基本構造と相性の悪いプログラムスタイルである.

そこでメインのプロセスは,単にほかのプロセスからのメッセージを待っていればよいようにする.ほかのプロセスがそれぞれに自分の担当の部分のイベント待ちをして,何かが起きたらメインのプロセスに伝えるわけである.イベント待ちをしているプロセスたちは,そのイベントが起きないかぎり何もしないので,CPUが常時何かをする必要は起きない.

このようなプログラムスタイルを実現するには,多くのプロセスからのメッセージを待ち受けられる機構があればよい.ここに,プロセス間通信と呼ばれる機構が使われる.単なるメッセージ伝達の道具として使われることもあるが,RPC(Remote Procedure Call)という,戻り値がある,関数呼び出しのような形で使われることが多い.これは本来ほかのコンピュータとの通信に使うために作られたインターフェイスだが,通信するプロセスにサービスとそのサービスを使うクライアントという役割を与えられる場合には,常に有効である.

Programming Paradigm of Window System

ウィンドウシステムが機能するためには,ウィンドウのそれぞれに独立したプログラムを走らせる必要がある.これには,当然プロセス間の通信機能が必要である.

UNIX環境で最も一般的なウィンドウシステムであるX Windowは,全面的にクライアント・サーバ・モデルにのっとって作られている.画面表示とユーザーからの入力の取りまとめを担当するサーバが,クライアントであるユーザープログラムからの描画要求を,RPCで受け取るという構造だ.

このような構造をとることによって,ウィンドウシステムで動くプログラムを,実際に表示を担当するハードウェアと独立させることができる.物理的に違うコンピュータで動いていてもよいわけであるし,デバイスに依存したコーディングの必要もなくなる.

ここで当然問題になるのは,このような手法によって効率が落ちるのではないかという問題である.しかし,直接グラフィックのハードウェアをいじるプログラムを書いたのでは,まず間違いなくアプリケーション間で衝突を起こしてしまうだろう.どちらにしろ抽象化したインターフェイスを使わないかぎり,多くのプログラマの手になるプログラムがデバイスを共有して同時に動くことなどはできない.これは,そもそもの最初にオペレーティングシステムが生まれた理由と同じことである.

すると問題として残るのは通信のオーバーヘッドである.このうち最大のオーバーヘッドは,オペレーティングシステム内部で発生する.これはプロセス間通信内部の機構によって発生するものでは

なく，単純にプロセスが切り替わるというだけの理由によるものである．

X Windowの場合だけでなく，アドレス空間が切り替わり，ディスパッチャがどのプロセスをディスパッチするべきか検討することによるオーバーヘッドは，ウィンドウシステムにとって非常に重要な問題である．このオーバーヘッドを減らすために，1つのアドレス空間の中に，複数のライトウェイトプロセスやスレッドなどと呼ばれる軽い処理単位を置くという改良も進められている．これらは，特定の状態が起きたときだけCPUにディスパッチされるようにプロセスの中で待たせておくことができるのだ．

Multiprocessing

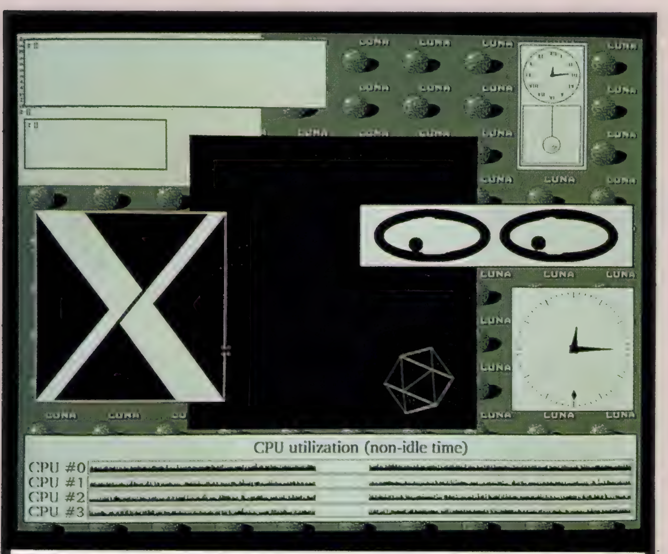
多くの仕事を多くのコンピュータにやらせるためには，OSに相応の調整機能が必要である．CPUに何を共用させて，どのように仕事を割り振るかが問題になる．

多くのプロセスが処理を進めている場合，処理を並列に行なって処理効率を上げるのはそれほど難しいことではない．同じCPUを複数用意すればよいのである．大型コンピュータなどでは，このようなシステムが一般的で，ある程度以上の処理性能を出すにはCPUを増やして並行に処理を進める．

このような処理を効率良く実現するには，高速でプロセッサの状態を切り替える機能や，割り込みを受けるCPUの固定，排他制御についての配慮など，さまざまな努力が必要である．大型コンピュータでは，このような努力も一般的であったが，比較的スケールの小さいオペレーティングシステムであるUNIXでは，同時に動くプロセスの数の少なさもあって，まだまだ一般的ではない．

ところが，ウィンドウシステムの項でも触れたように，プロセスを複数のスレッドに分割すると，本質的に1つの仕事であっても，複数のCPUに分けて実行することができるようになる．Machは，このスレッドを積極的に利用して，並列処理を実現するべく作られたUNIX系のオペレーティングシステムである．MachはUNIXに軽いスレッドと，スレッドやプロセス間の通信機能が強化されている．さらに複数のCPUを使いこなせるようにディスパッチャも拡張されている．またMachでは，オペレーティングシステム本体もいくつかの部品に分かれており，構成をシンプルにするとともに，オペレーティングシステム内部でも並列処理がやりやすくなっている．

さらに，1つの計算を並列に実行することができる場合がある．スーパーコンピュータが，ベクトル演算器によって大量の演算を同時にやるのもその一例だが，もう少し複雑なアルゴリズムを使えば，これほど自明には並列処理できない問題も並列処理できる．そのような処理のためのコンピュータの1つにNCUBEがあり，1つの計算を最大1,024台のCPUに分散して進めることができる．ただし，このようなシステムでは，プロセッサ間の通信機構がある程度特殊になり，オペレーティングシステムも変化する．



88000マイクロプロセッサを4個搭載し，Machオペレーティングシステムで動作している，LUNAワークステーション．X Windowの画面の下のほうには，各プロセッサの活動率のグラフが表示されている．

多次元の立方体の頂点にプロセッサを置き，辺にあたる部分を高速のプロセッサ間通信機構で結んだNCUBEコンピュータ．その結果最大1,024個のプロセッサを，並列に動作させることができる．ただし演算アルゴリズムには工夫が必要である．

debate

オペレーティングシステムとコンピュータ

スーパーコンピュータから大型汎用，さらにパーソナルなシステムまで，実際にオペレーティングシステムの製作と維持に当たっているエンジニアたちに，オペレーティングシステムのおかれている状況と今後の発展の方向性について語ってもらった．

最適化と抽象化

――OSの発展の方向性として，オブジェクト指向による抽象化がありますね．

堀：その方向はもちろんあると思います．ただ，どうしても最後に出てくるのは性能とのトレードオフになってくるんです．たとえばうちの半導体記憶装置，普通のファイルシステムとして使えるんですよ．そこにファイルをアロケートしたり，アクセスしたりできる．しかし，ほんとに最高性能を出すためには，そのリードライトのときのバッファのアドレスが4バイトバウンダリに合っているときが一番速い．そうすると**トランスペアレント**でなくなっちゃう．

――それでは，何でも4バイトバウンダリでアクセスしてればいいかというと，そういうものでもないですね．

堀：だから，4バイトバウンダリだとか8バイトバウンダリだとか，あらかじめ分かってる人はこっちを使う．で，分かんない人はこっちを使いますよ，というふうな選択になるんですよ．OSというのは最高性能を追求したい人は，まあそれなりの手段がありますよ．何も考えない人は普通に使っていいですよ．その2つの選択を用意しなきゃいけない．

海野：基本的には一番大きなOSはそうい

PROFILE

本格的なオペレーティングシステムのエンジニアは，数少ない．オペレーティングシステムのプログラムは，その並列性による難しさや信頼性，効率の追求により難しいのである．

●日本電気㈱ 堀 健一

現在，スーパーコンピュータSXシリーズのUNIX系OS，SUPER-UXのネットワーク系を製作している．以前には小型汎用コンピュータのオペレーティングシステム，ACOS2の中核の通信管理部分の製作にも携わった．

●日本IBM㈱ 海野三郎

古くからオペレーティングシステムの開発にかかわり，中型汎用コンピュータのOSであるDOSや，大型汎用コンピュータのOS，MVSに詳しい．また，パーソナルコンピュータや，オフィスコンピュータの経験もある．

●マイクロソフト㈱ 川合英俊

これまでマイクロソフトでMS-DOS，OS/2Version 1.1などのオペレーティングシステムの開発，さらにMS-Networksの日本語化などに中心的にかかわってきた．現在は，マイクロソフト製品の企画全般に携わっている．

う選択を追求すると思うんです．ただし，中規模になってくるとその両方の追求は難しい．

——オフコンなんかですと，抽象化できるだけ抽象化して，ユーザーは好きなようにアクセスしてればいいようにする．

海野：汎用コンピュータが選択可能で，オフコンは抽象化すると．では，もっと小さいパソコンはもっと抽象化を進めていいかというと，意外とそうでもない．

堀：逆に資源が小さいですからね．

海野：それにオフコンに比べるとソフトウェアベンダーが非常に力量を持ってて，マニア的なプログラミングをいとわない．そういう人に対して，非常に有効なインターフェイスを提供するっていうのも要求されてるんじゃないかと思いますが．

川合：私どものOS/2についてるHPFS，あれはオブジェクト指向ですね．

堀：HPFSというのはどういうものなんですか？

川合：ファイルシステムの中にB-tree，B+-treeの管理機構を組み込んでいるんです．で，その中にアイコンなどのアトリビュートのエリアを自由に取れるようになっています．

堀：そういったファイルシステムというのは，OSの中核の中になるんですか．

川合：普通のインストラブル・ファイルシステムですね．ファイルシステムとしてカーネルの壁の外にある．だから，これをさらに高度なものに入れ替えることは可能です．

堀：汎用機でも，マルチメディア思考っていうのはあるわけです．で，図形だとか，ビットマップだとか，グラフィックスだとか，音声とかそういうものをファイルの中に入れる．そのときにメインフレームがどうするかというと，普通のファイルがあって，その中の処理は，OSの外に置いたライブラリでやろうとするんです．それはインストラブル・ファイルシステムという考え方がないからなんじゃないかという気がします．

柔軟性と信頼性

——結局フレキシビリティとパフォーマンスという条件がでてくるんですね．

海野：もう1つ，パーソナルユースなどはいいとして，オンラインリアルタイムの処理でいつも要求される信頼性という条件があります．フレキシビリティが，OSの信頼性にある程度影響を与える．

——フレキシビリティが信頼性を下げるということですね．

海野：たとえば，インストラブル・ファイルシステムを提供するということは，どんなファイルシステムが入ってくるか分からないわけで，何かあったときのリカバリができるかどうかも分からない．そういう問題があると思うんです．ですから，ある程度以上OSの信頼性に影響するところは絶対に解放できない．

堀：そうですね，信頼性のために，一番低レベルのディスクI/Oドライバなんかは，ファームウェアレベルのリトライだとかOSレベルのリトライだとか，ものすごくいろいろやってますね．こっちのパスで失敗したら，5回まで繰り返して，だめだったらこっちのパスに切り替えて，ひょっとしたら反対側のIOPが生きてるかもしれないから，こっちでやってみると．それでもだめだったら，今度はマスストレージ・コントローラをひっくり返してやってみる．それでもだめだったら，一応カーネルに返してマスストレージ・コントローラのファームウェアをリイニシャライズしてもう1回やってみるとか．それでも死なない．死なないというか処理を中断しない．要するに，10日間動き続けて，あと1日で終わるジョブっていうのがそこで中断されてはたまらんということですね．もう1つは**トランザクション**で時間的な制約があり，かつ絶対落ちてはいけない場合．

——トランザクションの終了時刻の保証をするという問題があるんですね．

海野：どっかで処理が詰まると，雪だるま式にほかのトランザクションがみんなつっかえて，全体が動かなくなる．そういう意味では，会話モードではそれほど信頼性を吟味しないのは，倒れても割と早くリスタートしてくれればいいから．それに対して，トランザクションでは動き続けてくれなきゃ困るというので，OS

スーパーコンピュータ

スーパーコンピュータは，科学技術計算に特化し，1GFLOPSを超える計算能力を誇る．この速度でも計算に日単位でかかる，巨大なジョブを走らせることも多い．当然，ユーザーは性能の向上に大きな期待をよせることになる．また，いったん電源OFFしても処理を再開できるといった特殊な機能も要求される．

CPU ベクトルプロセッサを持ち，1命令1クロックで大量の演算を実行できる．CPU自体も複数組み込まれる．クロックは数百MHz．

メモリ 実メモリはGbytes単位で実装される．当然ディレイは非常に少ないものが使われている．

I/O 100Mbytes/secのインターフェイスに，インターリーブしたハードディスクや，大容量の半導体記憶などが接続される．

トランスペアレント　途中に介在しているものの影響が透明で見えないこと．この場合はデバイスを抽象化して使っても，最大の性能を引きだそうとすると，物理デバイスに依存した処理をせざるをえなくなるという意味で使われている．ファイルを通信を介してアクセスした場合と直接にアクセスした場合を比べるときなどにも使う言葉．

トランザクション　データベースを参照して更新する場合のように，完全に終了するか何も起こらないかのどちらかでなければならない処理．口座振り替えはその一例．もとの口座から必要な金額を差し引いたところでエラーを起こして止まってはならない．また，そのような場合に元の口座にその金額を戻す処理をロールバックという．

の機能の提供の仕方に差が出てくる．

堀：OS/２の資源管理には，あらかじめ資源を確保するという考え方はあるんですか．処理開始前に資源を確保して，全部確保できたからといって動き始めるという概念はあるんでしょうか．

川合：それぞれの資源を順次オープンしにいくかたちになります．

堀：で，だめだったら待つという概念ですか？　そうするとあっちでオープン，こっちでもオープンで，**デッドロック**が起きませんか？

川合：それは起きます．

堀：トランザクションの例では，ここから始まってここが終わるまで絶対に何秒以内というふうに決められていると，まず資源確保してからでないと，物事を始めない．明日の天気予報に３日かかっちゃ困るというようなことがあると，それを保証するための機構というのが要るんです．パーソナルユースでそういうのが必要になってくるかどうかっていうのは，よく分からないんですけど．

川合：そこまでの要求は出てこないでしょうね．パーソナルユースでは，それよりもレスポンスタイムのほうを要求する．

堀：大型機とパソコンの一番の違いは，持っているMIPSの何十%をユーザーインターフェイスに使ってるかということでしょうか．

海野：汎用コンピュータも非常にレスポンスタイムってのを気にしていて，CPUはどんどん使わせるけれども，常に割り込み可能な状況にしようということはしています．ただ，それを両立させるのは難しい．ディスクの入出力などはCPUの空き時間を待たずに進めて，CPUは極力効率的に使って，速く処理を終わらせようということですね．

堀：そこまでシビアではなくても，CPUがやるべきことがたくさんあるという感じがありますね．だからパーソナルコンピュータがキー入力をビジーウェイトで待ってるっていうのは画期的なインターフェイスの違いだと思いますよ．大型コンピュータの世界だと必ずエンターキーがいる．そのタイミングで割り込みを起こして動く．

リソースの限界

堀：パーソナルコンピュータでメモリが足らないだとか，何とかソフトのっけると何かがのらなくなるだとか，そういった話をよく聞くんですけれど，パーソナルコンピュータのレベルだと，メモリは足りな気味の資源なんですか？

川合：MS-DOSの場合ですと仮想記憶がのってませんから，640Kbytesにシステムが100Kbytes，あと常駐型のプログラムをいくつか取れば，アプリケーションには300Kbytes程度しか残らない．OS/２または今度のWindows3.0があると，仮想記憶が入ってるんで結構とれるでしょう．ただし，アプリケーションがこのセグメントはスワップアウトしないでねと，設定してしまうことができるんです．そうするとOSはぜったいにそのセグメントを外へ出しませんから，メモリがいっぱいになってしまう．

堀：ただ張り付ける．

川合：アプリケーションは自分のことしか考えずに，完全に張り付けですからね．システム全体を考えた場合には，スワップということを考えなきゃいけないんでしょうけれど，MS-DOSの環境でアプリケーション書いていたのを前提とすると，自分の環境はいつも決まったところにあるというふうに考える．

堀：大型システムはシステム全体のパフォーマンスを気にしますからね．私はメモリをこれだけ張り付けてくださいっていうのは相当非標準なことです．

――インターフェイスは用意されていて，その気になれば実メモリを何メガバイトでもとれますよね．

海野：そういうことを許すべきプログラムっていうのはある種の特定のプログラムという感じです．

堀：普通はやらない．

海野：ええ，ただ普通でないのが結構ありますけど．オンライン系は必ずやりますから．そうやってOSを機能強化していくのは，トレンドだとは思うんですけども，私はMS-DOSのシンプルさというのは非常に忘れ難いですね．いいものを持っていると思うんです．MS-DOSが１Mbytes以上のメモリをサポートして，ファイルI/Oをやってる最中にほかの仕事ができたら，それはまだまだいろんなことができるOSだろうと思います．

同期I/O

堀：MS-DOSって必ず同期待ちなんですか．

海野：そうなんです．要求したファイルのI/Oが完了するまで，アプリケーショ

大型コンピュータ

大型汎用コンピュータは，大企業の事務処理や銀行のオンラインなどのホストコンピュータとして使われる．数千人のユーザーが同時に使用するのが普通である．きわめて多くの処理を並行して進めることになるので，処理効率を上げることが重要になる．また，信頼性に対する要求も非常に高い．

CPU　クロック50MHz程度のCPUが，メモリを共有して動作する．命令セットは10進演算や文字列処理を含む，事務計算用の命令を含む．

メモリ　256Mbytes程度の実メモリを，多くのCPUで共有する．当然仮想記憶機構も持つ．

I/O　4.5Mbytes/sec 程度のチャネルに，最大4096個のディスクなど周辺が付き，これらすべてと同時にI/Oをすることができる．

海野三郎

ンはストップしてます.UNIXでもプロセスはファイルI/Oで止まりますよ.

堀：いや，UNIXのI/Oは基本的にキャッシュに書くだけです．だから，実際にディスクまでいっちゃうってことはない．ライト側は常に非同期だね．

海野：まあ，ただそれはキャッシュに書いて，書くというごまかしによって，非同期になるんであって，本質的には同期でしょう．

堀：だからシングルタスクで性能を上げたいっていう人は非同期のI/Oを使わなきゃいけないんでしょうね．で，マルチタスクにしちゃって，たとえばディスクI/Oの部分はそのタスクに全部任せちゃうっていうふうにすれば，タスク間通信が非同期I/Oだから．実質的に非同期I/Oになるんです．

——I/Oというものをどう抽象化するか．同期，非同期っていうだけじゃなくて，何かもうちょっと抽象的なインターフェイスがあってもいいんじゃないかという気がするんですけれど．

堀：それはひょっとしたら，アクセスメソッドというふうな言い方になるのかもしれませんね．VSAMはある程度高機能化した，バーチャルなインターフェイスに対するアクセスであるという言い方もできますか．

海野：ちょっとそこまで言えるかどうかは分からない．ストレージからみると，ファイルっていう物をあまり意識しないで使えるという可能性は出てくると思うんですけどね．今は半導体メモリをディスクキャッシュなどのかたちで使ってますけど，そうじゃなく，メモリというかたちで有効に使って，プログラムからファイルという意識はあまりしないでもかまわないというようなインターフェイスにしたい．

堀：それはマップドファイルとか，そういった感じになりませんか．

海野：マップドファイルもそのひとつの考え方だとは思うんですけれど，私が意図したのは，半導体記憶をうまく使おうという話で，ディスクコントローラがある程度サーバ的な機能を提供するというようなことです．ほんとにそこまでOSでやってほしいのかどうかというのは，ちょっと分からない状態ですけれど．

プロセッサ管理

——メモリとI/Oの話がでてきたので，3つ目の話題としてCPUの管理はどうでしょうか．

堀：スーパーコンピュータのベクトルレジスタっていうのはたくさんあるんですけれど，ベクトクルの長さ，本数が設定できるんです．プログラムごとに変えられる．そのプログラムに最適な長さのベクトルレジスタを使うんですね．

海野：CPUに関しては，主記憶装置を共有する**密結合**の範囲という区分けがあります．たぶんその区分けはよりあいまいな方向にいって，疎結合のCPUどうしでも，タスクを相互にディスパッチしあうというような方向はでてくるんじゃないかと思うんです．

堀：それは，ありますね．特にトランザクション系で，こっちのCPUが死んだらプロセスはマイグレーションして別のCPUで生き残るというような機能は必要ですね．

川合：パソコンのレベルでは，ストレージカップリングがやっと始まったぐらいですね．

堀：それはネットワークを通してのストレージの共有じゃなくって．

川合：バス経由で．

堀：SCSIバスに2つCPUを置く．そんなイメージなんですか．

川合：SCSIではなくて，EISAとかMCAとかですね．

堀：それで，共有するのはメモリですか，それともディスクなんですか？

川合：ディスクですね．

——スーパーコンピュータでは，どのCPUをどういうふうに使うとか，ユーザーから勝手に指定できるわけですか？

堀：ひとつのプログラムがたくさんCPUをとって走るっていうやり方があるんですよ．で，DOループの1/4はこのCPU，この1/4はこのCPUというふうに割り付ける．そういうときにはプロセッサ数を指定する．

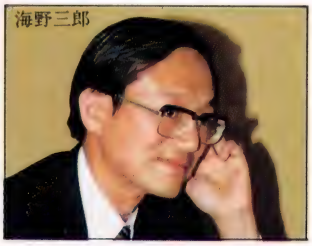
オフィスコンピュータ

COBOLによる使用のため最適化されたオフィスコンピュータ．用途的には大型コンピュータと似ているが，それほど多くのプログラムが同時に動くことはない．ユーザーは，それほどコンピュータに詳しくないため，ハードウェア，ソフトウェアのメンテナンス性のよさが重視される．また設計は比較的新しいものが多い．

CPU　処理性能的にはパーソナルコンピュータ並みから，大型コンピュータ並みまで幅広く分布する．複数CPUの製品も多い．

メモリ　数Mbytesから十数Mbytes程度が標準的．AS/400は，48bitの巨大なアドレス空間を持っている．

I/O　数百Mbytesから数Gbytesのディスクを接続することが多い．ディスクアクセスは中間的な15ミリ秒程度になる．



デッドロック　2つ以上のプログラムが，お互いに相手の必要とするファイルなどの資源を独占して放さず，処理が進まなくなること．2つのプログラムが同じ2つの資源の独占を必要とするのに，違う順番で1つずつ確保すると起きる．解決策は，あらかじめすべての資源を確保するか，あるいは資源を確保する順番を固定すること．

密結合　密結合マルチプロセッサ．複数のCPUがメモリを共有して動作する並列処理のアーキテクチャ．スーパーコンピュータ，大型コンピュータで使われる．疎結合マルチプロセッサは，複数のCPUがメモリを共有せずに動作する．今のところ電子交換機や，信頼性を重視するコンピュータ，並列計算機などに使われている．

——プロセッサ数を書くインターフェイスがちゃんと用意されてるわけですね．
堀：そう．書けるときもある．ただ，どのCPUかは見てない．
海野：ということは，アシンメトリックな構成ではない．
堀：カーネルは必ずこのCPUが知ってるとか，そういうのはあります．たとえば，タイマ割り込みは必ずこのCPUにしか来ないとか，I/O割り込みはここで必ず受けるとか，そういった機能差っていうのを設けることはできます．
——CPU間のコンテクストスイッチを最小にするように，アプリケーションでCPUの最適な割り当てをすることなどはできるんですか．
堀：それはやりますよ．アプリケーションがといっても，アプリケーションにくっつくサブルーチンがやるんです．そのサブルーチンの中でディスパッチャを持っていて，あるアルゴリズムでCPUを割り付ける．そういう意味でCPUにディスパッチするというところが，実は特権命令ではない．ちょっと特別な命令になって，カーネル内部じゃなくても実行できます．
——特権命令じゃないのに，CPUが制御できるわけですか．
堀：マニュアルを見るとおかしくって，特権命令とノーマル命令のほかに，特権ノーマル命令というグレーゾーンがあるんですよ．これがライブラリ・スケジューラの使う命令です．CPU切り替えるのに，カーネルまでいかないんですよ．カーネルにいくと1回例外起こすし，ぐじゃぐじゃといろんなところを走って，結構時間がかかるんですよね．それを嫌って，サブルーチンコールだけですましてしまう．
——そこまでやるんだったら，もうカーネルの中も外もない．
堀：だから，一応グレーゾーンがある．
——アプリだけあればいいんじゃないかということですね．
堀：スーパーコンピュータなんかね．フォートランさえ動きゃいいんだ，あれはフォートランマシンだっていう，そういう位置付けもありますからね．

名前の管理

——OS中でいろんな名前を使いますね．これをどういうふうに統合していくかという名前の管理の問題はどうでしょう．
堀：ツリー構造の名前空間っていうのはある程度直感しやすいし，管理もしやすいということで，わりとよく使われてると思うんです．特にUNIXだと，ファイルとはとても思えないようなものも，一応ファイルとして実現してしまうことによって，ユニークな名前を実現してます．OS/2ではそういう名前空間の流用といったものはあるんですか？
川合：まあ，ファイルは基本的にツリー構造で管理してますよね．ただ，それをどこまで拡張するかですね．
堀：ファイル以外のものに，ファイル名を付けちゃうっていうこと．
川合：スタンドアロンでもデバイスとか，ネットワーク上で先につながっている場合にもファイル名を付けています．
堀：それはファイルシステムの中の名前として決めるんですか．
川合：ファイルシステムといいますか，システムの中のデバイス名になるんです．
堀：それは名前空間の種類としては全然別の空間に属する名前なんですか．
川合：システムが管理しているエリアではありますね．特殊な名前のリストっていうのがMS-DOSと同じようにリザーブされてます．
堀：ACOS 2のファイルシステムは，実はMVS互換なんですよね．
——あれはネーミングはどうなんです？
堀：ファイル名，データセット名っていうのがあって，それとは全然別に**カタログ**っていうのがあるんです．
——カタログ名っていうのとボリュームシリアル名っていうのと，ジョブ名．名前空間がいっぱいありますね．
堀：特にカタログっていうのは，いかにも後からとってつけたという感じがしますね．
海野：しますね．ただ，今はほとんどカタログで判断するんですよ．
——カタログという1階層抽象化された名前で，全部扱えるようにしようということですか．
海野：いや，そこまでの抽象化はしてないんです．ただ，ボリュームシリアルというデバイス名があって，そのボリュームシリアルが違えば同じファイル名が幾つもあってもいいわけです．少なくともそういうファイルはカタログに一意の名前で入れることができる．カタログが違えば同じ名前もありうるわけですが，基本的にはそれも対処して，名前ですべて管理しようという方向にはいってると思

パーソナルコンピュータ

一般のオフィスの机上，家庭などで個人的に必要とされる計算処理をするためのシステム．非常に多くの種類のアプリケーションが使用される．システムがダウンしても代替機があることも多く，信頼性はそれほど重視されない．また，ユーザー管理の機能は持たないのが特徴である．

CPU 16bitから32bitのマイクロプロセッサ1個で構成されていることが多い．処理能力は大きいが，アドレス空間などに制限がある．
メモリ 256Kbytesから3Mbytes実装すれば，基本的なOSが走る．最大実装は16Mbytes程度．
I/O 20Mbytesから100Mbytesのディスクが最大で15個程度つなげる，1.2Mbytes/sec程度の速度のインターフェイスを持つ．

川合英俊

うんです．ただ，それはあくまでもデータセットの空間であって，名前の空間を1本にしようという話ではない．

——UNIXの場合にしろMVSの場合にしろ，デマウンタブルなファイルシステムがあって，その上に何か名前があったら，その名前は一体どうなるんだろうかという問題があるんですが．

堀：カタログ上にはあるんだけど，実体はないとか．

海野：その場合はマウント要求を出すということになりますね．

堀：OS/2にはマウント要求という概念がないですよね．オートマチック・ボリューム・リコグニション，つまりファイルにラベルが付いていてマウントしたときにOSが認識するっていうふうな機構もないですよね．

——Macintoshには，非常にはっきりした機構がありますね．

川合：FATのファイルシステムを拡張したときに，フロッピーディスクにシリアルナンバーを付けるという形態にしてます．

堀：それは，どんなきっかけでシステムの中に取り込まれるんでしょう？　フロッピーディスクをガチャンと装置にマウントした時点ですか．

川合：その時点では何もしない．何らかのオープンの要求があって，そのファイルが開けられたときにフロッピーが入れ替わっていれば，同じファイル名のものがあっても別のファイルだと認識する．

OSの問題点

——今のOSの抱えてる問題点には何があるでしょうか．

堀：まず，大きくなってきたっていうのが1つあります．OSというのは複雑なんですよ．OSのマニュアルを買うと，ドーンと大量にやってきてうろたえてしまうという話がありますね．

海野：問題点って言われると，あまりにもいっぱいあって多過ぎて言えないけれど，汎用機の場合の一番の苦しさは互換性を保ちつつ発展させるところが，一番苦しいところだと思うんです．互換性を断ち切ってやっていくとすると，かなり思い切ったことでもできると思うんですが，ここ10年間誰も触ってないジョブなんていうものがいっぱい流れてますので，そうはいかない．互換性を保ったうえでどう改善していくかというのは最大の問題なんですね．

堀：とにかくユーザーのソフトウェア資産は絶対ですからね．何十億，何百億というインベストメントがあって，それを裏切るわけにはいかない．どんな昔のバージョンのロードモジュールでも絶対走るっていうのが至上命令ですね．

——スーパーコンピュータは，互換性の

面では，あまり心配しなくてもよかったのではないかと思うのですが．

堀：それは確かにあるんですよ．スーパーコンピュータは，互換性はある程度犠牲にしても，性能のほうを追求するということですから，トレードオフのポイントがちょっと違うんですね．

海野：それは性能が非常に高ければユーザーも許してくれるけど，たかだか2倍ぐらいで互換性をなくしたらだめですね．

川合：OS/2で一番苦しんだのは，やはりMS-DOSのアプリケーションがそのまま動くようにすることでした．

——その互換性っていう条件をはずしたら，大型コンピュータのOSはどんなものになるんでしょうか．

海野：大型コンピュータの互換性を外したら，OSの前にハードウェアのアーキテクチャも変えると思うんですね．かつて汎用コンピュータは8ビットデータから64ビットデータまで互換性を持たせる命令セットを作ったわけですけど，おそらく64ビットデータしか前提にしないアーキテクチャにするだろうと思います．

——**ワードマシン**の方向に振っていくということですか．

海野：いや，バイトマシンのまま命令長を一定にするとか，今のRISCのCPUで持っているような機構を取り込んでいく．また，今はOSの中にバッチ処理の機構があって，オンライン系はその上に乗っかってますけど，それを一体化できると思います．

——ファイルシステムだとか，メモリまわりではどうでしょう．

堀：それはいっぱいあります．やっぱり昔のファイルは読めなきゃいけない．

海野：くだらない話ですけど，今ページング4Kbytesでやってますが，これは今の主記憶装置が256Mbytes，1Gbytesに迫ろうかというときにあまりにも小さい．

堀：ページテーブルもいっぱいいる．

——大型コンピュータとしてはだいたいそんなところですか?

海野：作り直せたらどこに進出するかという自由度はいろいろありますが，絶対残るのは，高信頼性と性能を最大限に発揮するというところなのは間違いない．

堀：互換性っていうのは時間軸上で，過去との整合性を保つことなんですけどね．コンピュータネットワークっていうのが制約になってしまうかもしれない．他システムとの整合性を守らなければならない．もう外から見た場合のビヘイビアは絶対プロトコルどおりでなければいけない．これもOSを作っていくうえでの制約条件の1つでしょうね．それはデータ交換のためのファイルフォーマットだとか，MT上のファイルフォーマットだとか，ラベル形式とか，そういうのすべて含んでのことです．OSはこれからもその制約の上で発展していくんでしょうね．

カタログ　MVSにおいて，ファイルは本来ボリューム名と，そのボリューム内のVTOC (Volume Table Of Contents)にあるファイル名のリストによって管理される．ファイルをそのボリューム名と対応付けるリストがカタログである．ただし，システムにカタログは1つとは限らず，カタログのあるボリューム名を指定することもできる．

ワードマシン　現在のプロセッサの主流は，バイト単位でプログラムやデータをアクセスするバイトマシンである．これに対して，ワード単位でのみメモリをアクセスするプロセッサがワードマシン．現在はLISPマシンなどに使われているくらい．Pascalの設計には，当時のターゲットがワードマシンであったことが影響を与えている．

OSとMS-DOSはどこが違っていたか

MS-DOSに不足している機能，あるいは一般のオペレーティングシステムには用意されているのにMS-DOSにはない機能と，その機能がどのような状況のもとで必要であるのかは，おおよそ分かっていただけたものと思う．プログラムを書くうえでの苦労の幾分かが，オペレーティングシステムに機能が欠けていたせいであったということに，あらためて気付かれた方も多いのではないだろうか．

しかし，オペレーティングシステムに機能を追加するのは，巨大なアプリケーションに機能を追加していく以上に危険なことである．いったん動き出したオペレーティングシステムが小さくなった例は，いまだかつてない．また，基本がシンプルなシステムに複雑な機能を追加して，その機能自体はうまく動いているように見えても，実はシステム全体の信頼性が下がった例は多い．機能追加にはバランスについての注意が必要である．

特にマルチタスクのシステムでは，排他制御が非常に難しい問題になってくる．デッドロックやシステムのクラッシュの可能性を下げるためには，かなりの注意が必要である．しかし，このような並列的な動作の管理は，一般的なシステムではオペレーティングシステム内部に特有のものであり，オペレーティングシステムのエンジニアだけの楽しみということができるだろう．

しかし，現実に使われているオペレーティングシステムに手を入れるとなると，楽しみばかりを追求するわけにもいかない．基本的な性能が重要である．オペレーティングシステムの性能は，単に処理効率だけではなく，信頼性，柔軟性が非常に重要である．オペレーティングシステムが不安定では，コンピュータシステム全体が信用できなくなってしまう．またハードウェア構成の変化への対応能力，機能の入れ替えなどの柔軟性も，オペレーティングシステムが使われる環境が千差万別である以上軽視できない．

大きさは，このような性能のすべてに悪影響を及ぼす．巨大なオペレーティングシステムは，それだけで信頼性にとっての不安材料であり，メモリの必要量が多く，オペレーティングシステム内部で消費されるCPU能力が増える．また，機能を追加するのは，それまでに追加された機能が多ければ多いほど難しくなる．

しかし，ウィンドウシステムなどのプログラムを実現するために必要な機能は多く，オペレーティングシステムは肥大する傾向にある．このような状況を解決する方法は1つしかない．新しいコンピュータのあり方に対応した，新しい思想のオペレーティングシステムを作ることである．過去のオペレーティングシステムの改良や互換性を保ったうえでの機能追加にとどまらず，新しいオペレーティングシステムが求められている．

オペレーティングシステムが，ある程度動き出してから成熟するまでには10年，20年というスパンの時間がかかる．次の世代のエンジニアのためにオペレーティングシステムを作り出すべきときがすでに来ているのである．

さて，今回の特集のテーマとしたオペレーティングシステムであるが，ネットワークとの関係はますます重要になっている．近ごろ，ネットワークとの関係でオペレーティングシステムに追加された機能は数多い．次号の特集では，このような必要が生じた理由である，LAN (Local Area Network)技術について紹介し，さらにネットワーク利用の最大のメリットであり，かつ困難なポイントである異機種間，遠隔地間のLANを相互接続する実験を当編集部を舞台に展開する．ご期待いただきたい．

参考図書

オペレーティングシステムの教科書は多いが，この特集の参考とさせていただいたのは，

■岩波講座ソフトウェア科学6 オペレーティングシステム，前川守，岩波書店，1988

さらに，オペレーティングシステムの具体的な実現方法については，プログラムソースにあたるのが一番である．UNIXに挑戦するのもよいが，最近のバージョンではかなり巨大になり，また以前ほどシンプルでもない．ソースの入手も難しい．そこで，

■MINIXオペレーティングシステム，タンネンバウム，アスキー，1989

機械可読なソースは，ディスクで別売りもされているが，商用パソコン通信ネットワークでも配布されている．最後に，オペレーティングシステムにある程度詳しい方は

■Self-Assesment Procedure XX Operating System, Communications of the ACM, February 1990

をやってみられるとよいだろう．学べることがあるはずである．

宛先

この特集に対するご意見，ご要望，ご指摘などは，こちらの宛先までお寄せください．今後の特集題材の選択についてのご意見もお待ちしております．

〒107-24
東京都港区南青山 6-11-1
スリーエフ南青山ビル
株式会社アスキー
スーパーアスキー編集部
特集の係

INSIDE SOFTWARE

32bit環境に対応した OS/2 Ver.2.0

この年末から来年に向けてリリースが予定されている，OS/2 Ver.2.0(以降OS/2 2.0)の仕様が公開され始めた．OS/2 2.0は，OS/2の32bit対応版として，かねてよりその動向が注目されていたOS/2の新バージョンである．MS-DOS，Windows，UNIXなど，混迷するOS環境の中で，OS/2はどのような役割を果たすのだろうか．

今回のInside Softwareでは，米国IBM社発行の『IBM Personal Systems Developer』(1990，Spring)誌に公開されたOS/2 2.0のメモリ管理機構，プログラムの開発環境などを中心に紹介する．

OS/2 2.0の概要

OS/2 2.0は，単にOS/2 Ver.1.x (以降OS/2 1.x) を32bit化したものではない．OS/2 2.0では，今後予想されるハードウェア環境やソフトウェア環境をにらみ，非常に戦略的な変更が加えられている．そういう意味でOS/2は，このOS/2 2.0でまったく新しく生まれ変わったといっても過言ではないだろう．そこでまずは詳細を述べる前に，OS/2 2.0の基本思想とはいったい何なのか，またそれに従い，Ver.1.xからどのような点が大幅に改良されているのか，といったことを簡単に紹介しておこう．

OS/2 2.0の基本思想

1987年4月にOS/2の第1バージョン(Ver.1.0)が発表されて以来，Ver.1.1(PMをサポート)，Ver.1.2(HPFSをサポート)と，OS/2 1.xがターゲットとしたのは286のプロテクトモードであった．つまりOS/2 1.xがめざしたのは，286のプロテクトモードを効率よく使用するオペレーティングシステムだということになる．このためOS/2 Ver.1.xのアーキテクチャは，セグメントアーキテクチャなどに代表される286の長所を生かす半面，286の制限(アーキテクチャ的な限界)をそのまま背負う形となっている．

OS/2 2.0では，1プロセス単位ではセグメントアーキテクチャが採用されていない(後述の「メモリ管理」参照)．メモリ管理機構やプログラムの構造など，OS/2 1.xにおけるアーキテクチャの“鍵”となっていたセグメント機構を使用していないのだ．OS/2 2.0のプログラムは，メモリをセグメント単位にこまぎれに使うのではなく，広大なメモリ領域を，あたかも自分が占有しているように，直線的に使用できる．このためOS/2 2.0のプロ

Copyright(C)1990 by International Business Machines Corporation
Reprinted by permission from IBM Personal Systems Developer, Spring 1990
pp16-22 “OS/2 2.0 Overview” by Michael S. Kogan
pp23-30 “OS/2 2.0 32-Bit API” by Michael S. Kogan
pp31-35 “OS/2 2.0 Tools and Program Development” by Michael S. Kogan, Robert Tycast

グラムの構造は，UNIX環境のプログラムなどと非常に似たものになる．UNIXからOS/2へ，あるいはOS/2からUNIXへとプログラムを移植するのは，それだけ容易になる．

OS/2自体も，80x86に依存しないよう配慮しながら再設計されている．これは，Intelの80x86以外のシステムにも，OS/2がインプリメントされる可能性を感じさせる．RISC OS/2への布石と考えてもよいだろう．このようにOS/2 2.0が目標とするのは，386に依存したオペレーティングシステムではない．それはあくまでも，“32bitに対応したOS/2の386バージョン”である．

OS/2 2.0で利用可能なプログラム

OS/2 2.0では，3種類のプログラムが実行可能である．1つはOS/2 1.x互換の16bit OS/2プログラム．もう1つはDOSアプリケーション．そして最後が，386のメリットを引き出す32bit OS/2プログラムである．これら3種類のプログラムは，EXEファイルのフォーマットによって区別される．

OS/2 2.0では，OS/2 1.x互換の実行環境が用意されているため，1.x用のプログラムをほぼ完全に動作させることが可能だ．DOSアプリケーション，1.x用のプログラムともに，行儀のよいものならPMのウィンドウ内で複数同時に実行させることができる．ただし，これらのプログラムは，OS/2 2.0の全機能を利用することはできない．

OS/2 2.0で新たに追加されたのが，32bitモードである．APIの設計方針やPMの仕様などではOS/2 1.xを踏襲しているが，実質的には，新しい実行環境が付加されたといってよい．OS/2 2.0の全機能を利用するには，プログラムをこのモード用に作成する必要がある．

メモリ管理

OS/2 1.xでは，セグメントをメモリ管理の基本単位としていた．たとえば，n bytesのメモリをシステムに要求すると，大きさnのセグメントが用意され，そのセレクタが返される．プロセス間で共有メモリを使用する場合の共有単位もセグメントである．

このため，OS/2のプログラマは，常にセグメントを意識していなければならない．現在，プログラムの開発言語にはCを使うのが一般的だ．しかしCは，もともとセグメントという概念のないシステム用に設計された言語であり，Cとセグメントマシンとは相性の悪い面がいくつかある．

また，286ではセグメントレジスタの書き換えが比較的高価な（実行時間のかかる）命令であるため，セグメント単位でメモリを管理しているOS/2では，そのオーバーヘッドを無視できない．

周知のとおり，OS/2は仮想記憶をサポートしている．Ver.1.xでは，この仮想記憶もセグメント単位で行なっていた．これも性能低下の一因である．まず，セグメントの最大サイズは64Kbytesであり，1度のデータ転送量が多くなりがちである．また，個々のセグメントはサイズが異なるため，メモリ中のフリー領域が分断されやすい．それを回収するために，OS/2 1.xではメモリ中でセグメントのコピーを行なっている．特にこのセグメントのコピーにかかるオーバーヘッドは，システムの性能低下の大きな要因になっている．

OS/2 1.xにおいて，セグメントが問題となる理由がもう1つある．286CPUでは，1セグメントの最大サイズが64Kbytesに限定されていることだ．これは，現状のアプリケーションにとっては，あまりにも小さい．64Kbytesを超えるデータを扱うためには，複数のセグメントにデータを分散させなければならない．

OS/2 2.0では，プログラマがセグメントの概念を無視できるように，“フラットモデル”と呼ばれるメモリモデルが採用されている．フラットモデルのプログラムは，メモリ空間を広大な1つのセグメントとして扱う．つまり，コードおよびデータが，区別されずに同一空間に配置される．これは，従来のメモリモデルでいえば，DOSのCOMモデルに近い．プログラムではセグメントレジスタを操作する必要がないので，従来のように“64Kの壁”に煩わされることなく，プログラミングが可能である．

このようなモデルを採用できたのは，386では1セグメントの最大サイズが4 Gbytes（＝4,096Mbytes）と，286の65,536倍に拡大されているためだ．ミニコンやワークステーション系のシステムと同じメモリモデルが採用されたことで，相互のプログラムを移植する作業は，非常に容易になると思われる．

OS/2 2.0では，スワップもセグメント単位ではなく，386の機能を利用してページ単位に行なわれる．すべてのスワップ処理が，ページ単位，つまり4Kbytesの固定長で行なわれるためスラッシングが発生せず，メモリ内部のコピーは不要になる．これによって，メモリ管理のオーバーヘッドは大幅に抑えられる．

APIコール

従来のOS/2 1.xでは，API呼び出し時のパラメータを，いわゆるPascal呼び出し手順に従って行なっていた（パラメータを左から順にスタックに積む）．これに対しOS/2 2.0のAPI呼び出しでは，Cの呼び出し手順に従ってパラメータ渡しを行なう（右から順にスタックに積む）．Cの呼び出し手順と唯一違う点は，API名の前に‘_’（アンダースコア）を付加しないことである．

また，OS/2 1.xではセグメント間コール（farコール）だったAPI呼び出しも，OS/2 2.xでは32bitオフセットを使用したnearコールに変更されている．これによってセグメントレジスタの書き換えは不要になるので，APIコールにおけるオーバーヘッドも減少する．

OS/2 2.0では，386およびOS/2 2.0に対応した，新しいCコンパイラが提供される．このコンパイラは，386の32bit命令を生成するほか，OS/2 2.0のAPI呼び出し手順に従ったコール命令を生成する．コンパイラにOS/2 2.0のAPIコールである

ことを宣言するには，新しいキーワード“_syscall”を使用すればよい．

APIは，原則として2セット用意されている．1つは従来のOS/2と互換性のある16bit APIであり，もう1つは386の機能を最大限に引き出せる32bit APIである．OS/2 2.0対応のプログラムでは，1つのプログラムから両方のAPIを使用することができる．ただし，今後のリリースに対する互換性を考えるなら，32bit APIのみを利用することが望ましい．

VIO／KBD／MOUの各サブシステムは，16bit APIでしかサポートされない．これらのサブシステムは，PMが標準環境とされた現在では，不要と判断されたものと思われる．今後のユーザーインターフェイスは，PMによって実現されることになる．PMのAPIには，大きな変更は加えられていない．

カーネルAPI (Dos～) には，大規模な変更が加えられている．16bit APIと区別するため，API名は“Dos～”から“Dos32～”に改められた．ただし，このAPI名の違いはヘッダファイル(OS2.H)の中で吸収されているため，プログラマは従来どおりに“Dos～”を使用すればよい．

パラメータとして渡すWORD値はDWORDに，ポインタ値もfarポインタではなく32bitのnearポインタに置き換えられた．これは，フラットモデルに対応した変更である．DosReadやDosWriteなどでも，64Kbytesを超えるデータを一度に扱うことができる．大量のデータを扱うようなプログラムの開発や移植は，非常に楽になるだろう．

DOS互換環境

386の強力な機能として，仮想86モードのサポートを挙げることができる．これを利用すれば，複数の安全な86環境を実現できる(図1)．これは，仮想86モードでも386の保護機能やメモリ管理機能が働くためである．したがって，たとえアプリケーションが暴走しても，全システムが破壊されることはない．OS/2では，複数の仮想DOSマシンを同時に起動でき，しかもPMのウィンドウ内で実行できる．さらに，仮想DOSマシンの画面イメージ(テキストやグラフィック)を，クリップボード経由でPMアプリケーションに転送することもできる．

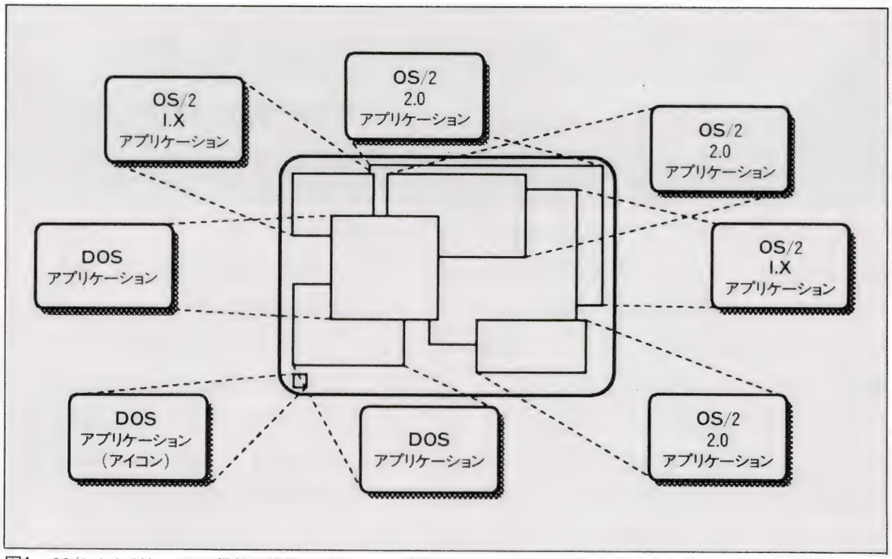


図1　OS/2 2.0では，386の仮想86機構を利用して，複数のDOSセッションを同時に実行できる．

また，DOSカーネル(MSDOS.SYSやIBMDOS.COM)に相当する部分は，386モード下で動作するため，仮想DOSマシン内のフリーメモリは，今までのDOSよりも広くなる．OS/2 1.xのように，メモリ量の制限による互換性の低下を心配する必要はない．なお，仮想DOSマシンでは，EMSやXMSのエミュレーション機能もサポートされる．

NDPエミュレーション

DOSやOS/2 1.xには，NDP(Numeric Data Processor：8087や80287などの数値演算コプロセッサ) のエミュレーション機能はない．そのため，実数演算を行なうアプリケーションは，演算ライブラリをプログラム内部に抱え込み，実行時にNDPの有無をチェックし，NDPを使うか，演算ライブラリを使うかを選択する必要があった．

OS/2 2.0では，NDPエミュレーション機能として，80387相当の演算ルーチンがOS内部に用意されている．アプリケーションからは，どのシステムでもNDPが使えるようにみえるので，演算ライブラリを内部に抱え込む必要はない．

デバイスドライバ

OS/2 1.xでは，全デバイスドライバがバイモーダル（リアルモードとプロテクトモードの両方で実行可能）でなければならなかった．この点が，ドライバの開発を困難にし，また実行効率の低下を招いていた．

OS/2 2.0のデバイスドライバは，常にプロテクトモード下で呼び出される．メモリモデルはOS/2 1.xと同様に，セグメント：オフセット形式のものなので，OS/2 1.x用のデバイスドライバはそのまま利用できる．

複数のDOS互換環境をサポートするために，従来のデバイスドライバはPDD (Physical Device Driver：物理デバイスドライバ) とVDD (Virtual Device Driver：仮想デバイスドライバ) の2つに分割された．PDDは従来のデバイスドライバに相当し，デバイスの管理を行なう．VDDの役割は，仮想DOSマシン上のプログラムが直接アクセスしようとするデバイス（シリアルポートやビデオボードのレジスタなど）のエミュレーションを行なうことである．VDDは，32bitアドレス環境で動作する．（梅原　系）

OS/2 2.0のメモリ管理機構

Michael S. Kogan

OS/2 2.0は,32bit CPUである80386の機能をサポートする,OS/2の最新のバージョンである.利用環境として,386DX／SX,および486を前提にしている.32bit環境をサポートするために,新しいプログラミングモデルを採用し,デマンドページングによる仮想記憶もサポートした.32bitプログラミングモデルでは,アプリケーション,サブシステム,カーネルのすべてで32bitレジスタと,32bit用の命令やアドレッシングモードを利用でき,64Kbytesを超えるメモリオブジェクト[*1]を自由に扱うことができる.

OS/2 2.0は従来のバージョンとも互換性があり,OS/2 1.x用のアプリケーションやサブシステムを実行することができる.また,386の持つ仮想86モードの機能を利用して,複数のDOSセッションをバックグラウンドで同時に実行することもできる.システムに,387NDP(数値演算コプロセッサ)のエミュレーション機能が内蔵された.ファイルシステムとしては,OS/2 1.2と同様,FATファイルシステムとHPFSの両方がサポートされる.

32bit プログラミング環境

32bitプログラミング環境の長期的な目標は,シングルプロセッサ構成のIntel 80x86マシンだけでなく,他のシステムに対してもポータビリティを持った,プログラミング・プラットフォームを提供することである.この目標を満たすために,“フラットモデル”(flat model)と呼ばれる,新しいメモリモデルが用意された.フラットモデルでは,メモリを32bitオフセットによってアクセス可能な,広大な単一のセグメントとして扱うことができる.

フラットモデルは,プログラムの移植性に寄与する.というのも,各プロセスのメモリ空間の先頭を指すベースレジスタと,その空間内のオフセットを指すオフセットレジスタを備えたハードウェアなら,容易にフラットモデルを実現できるからだ.つまり,OS/2 2.0は386に依存したOSではなく,数多くの“32bit OS/2”ファミリの中の386対応バージョンである,ということができる.

フラットモデル

386対応のOS/2で採用されたフラットモデルは,“0：32モデル”と呼ばれることもある.これは,32bitのオフセットアドレスだけで特定のメモリをアクセスできることを意味する.これに対して,OS/2 1.xは“16：16モデル”と呼ばれる.16：16モデルでは,メモリのアクセス時に16bitのセグメントアドレスと16bitのオフセットアドレスを指定する必要がある.

フラットモデルではセグメントの概念が隠されており,プログラマはセグメントの存在を意識しなくてもよい.この結果,セグメントベースのシステム以上の性能と,他のシステムとのプログラムの互換性の両方を実現することができた.

OS/2 1.xにおけるプロセスのメモリ空間は,LDT (Local Descriptor Table)に登録されたセグメントの集合体である.それに対してOS/2 2.0のメモリ空間は,リング3に配置された単一のセグメントであり,32bitオフセットでアクセスされる.フラットモデルでは,アロケーションの基本単位は4Kbytesの“ページ”になる.OS/2 1.xのセグメントに代わって,メモリ空間にはメモリオブジェクト(1つ以上のページからなる)が配置される.セグメントと異なり,メモリオブジェクトはリロケータブルではない.また,アロケートの基本単位は4Kbytesである.そして,メモリオブジェクトはプロセスのアドレス空間の4Kbytesバウンダリに合わせられる.

OS/2 2.0におけるプロセスのメモリ空間の大きさは,512Mbytesである.この制限は,将来のリリースでは4Gbytes (386の限界) までに緩められる予定である.512Mbytesという現在の制限は,OS/2 1.xとの互換性に由来している(詳しくは後述).フラットモデルのメモリイメージを**図2**に示す.

図中で,高位アドレス空間にシステム領域が確保されていることに注意していただきたい.アプリケーションが512Mbytesより高位の領域や,アロケートされていない領域をアクセスすると,アクセスフォールトが発生する.プロセスのアドレス空間は,OS/2 1.xのLDTと同様,プライベート領域と共有領域に分かれている.共有領域は高位アドレスから低位アドレスに向かって,プライベート領域は低位アドレスから高位アドレスに向かって順にアロケートされる.プロセスの起動時には,プライベート領域と共有領域のサイズとして,それぞれ最低64Mbytes用意されていることが保証されている.

メモリ保護は,16bitセグメントモデルとフラットモデルで,大幅に変更された部分の1つである.16bitセグメントモデルでは,セグメント単位でのメモリ保護を行なっていた(各セグメントセレクタがロードされるたびに,アクセス権とセグメント境界のチェックが行なわれる).

これに対して,フラットモデルでは全メモリオブジェクトが同一のセグメントに格納されている.つまり,セグメント単位の保護機能は利用されず,ページ単位の保護によって,メモリ保護を実現している.

フラットモデルでは,メモリオブジェクトをアロケートするときに,その領域中のページ属性を制御することができる.また,メモリオブジェクトを含むページのそれぞれについて,アクセスを許可す

＊1　OS/2におけるメモリ管理の基本単位.DosAllocMemによって確保するメモリ空間.

るかどうかを指定できる．もし，アクセスを禁止したページがアクセスされると，アクセスフォールトが発生する．コミット (Commit) やデコミット (Decommit)[*2] といった属性の変更などは，すべてページ単位で行なわれる．

アプリケーションやライブラリ内部のメモリ管理ルーチンのために，システムはページサイズ (386では4Kbytes) をエクスポートしている．**表1**は，各ページあるいはメモリオブジェクトが配置されているページグループ[*3]に対して設定できる属性の一覧である．

386の場合，EXECUTE属性とREAD属性は等値であり，WRITE属性を指定した場合はREADとEXECUTEも指定されたことになる（つまり，読み出し可能なページは，常に実行可能である）．GUARD属性は，スタックのオーバーフローチェックや，その自動伸長を容易にするための機能だが，アプリケーションに応じてそれ以外の目的に利用してもよい．メモリオブジェクト中では，コミットページとデコミットページを混在させることができるので，“希薄な”(sparse) メモリオブジェクトを作成することもできる．

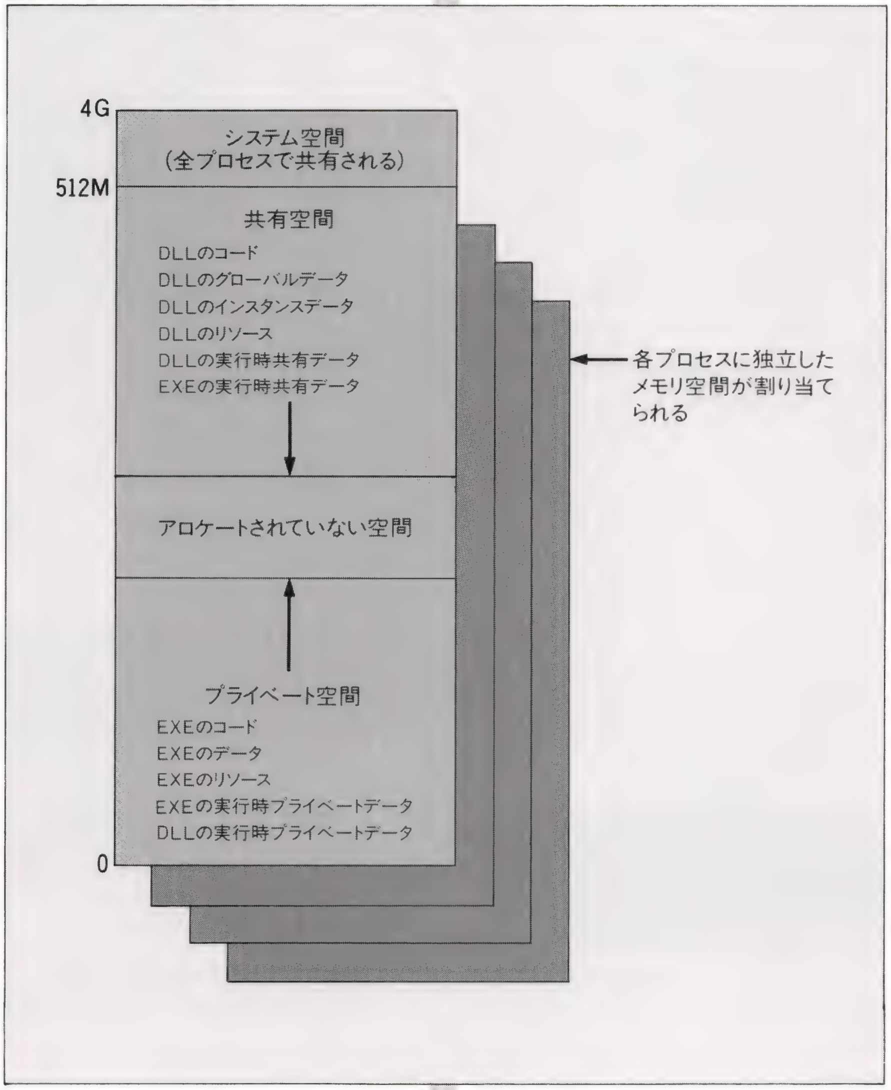
フラットモデルで作成されたアプリケーションやサブシステム，カーネルが，セグメントモデル（16：16モデル）よりも高性能なのにはいくつかの理由がある．その1つは，セグメントレジスタへのロード命令がなくなることだ．80x86のプロテクトモードでは，セグメントレジスタへのロード命令は，非常に時間のかかる処理である．フラットモデルでは，全アドレスが32bitのオフセットでアクセス可能であり，セグメントレジスタの書き換えを必要としない．実際，フラットモデルのプログラムは，セグメントやセグメントレジスタの存在を意識する必要はない．

フラットモデルのもう1つの利点は，プログラムのメモリモデルが1つだけですみ，64Kbytesの制限からも解放されるということだ．OS/2 1.xなど，8086や286をターゲットにしたシステムでは，数多くのメモリモデル (small, medium, large, hugeなど) を使い分ける必要があった．このほか，386の32bitレジスタや，32bit演算命令を利用できることも，プログラムの性能向上に寄与する．

表1　ページに設定できる属性一覧

COMMIT	ページは，特定のメモリ領域に対応している
DECOMMIT	ページはコミットされておらず，アクセスするとアクセスフォールトが発生する
EXECUTE	コードとして，命令のフェッチが許可される
READ	ページからの読み出しが許可される
WRITE	ページへの書き込みが許可される
GUARD	ページは保護されている

図2　プロセスのアドレス空間



＊2　OS/2 2.0で，ページ単位に指定できる属性の1つ．OS/2 2.0では，メモリ空間を確保すること (DosAllocMemによる) と，確保したページを実際のメモリ領域に対応させること（データやコードを実際に保持すること）を分離して管理できる．つまり，メモリ空間を確保しても，その空間にメモリが対応していない，という状態を作り出すことができる．これによって，大量のメモリを必要とする場合でも，まずはメモリ空間の確保のみを行なっておき，必要な場合のみ実際にメモリを対応させることができる．
　コミットとは，特定のページがメモリに割り当てられている状態を，デコミットとは割り当てられていない状態を意味する．

＊3　いくつかの連続したページをまとめて扱うための方法．OS/2のメモリ管理はページを基本単位にしているが，これは4Kbytesの固定長であり，プログラム上で扱う論理的なデータ（メモリオブジェクト）のサイズに一致しているわけではない．ページグループを利用することで，メモリオブジェクト単位での属性変更などが容易になる．

ページング

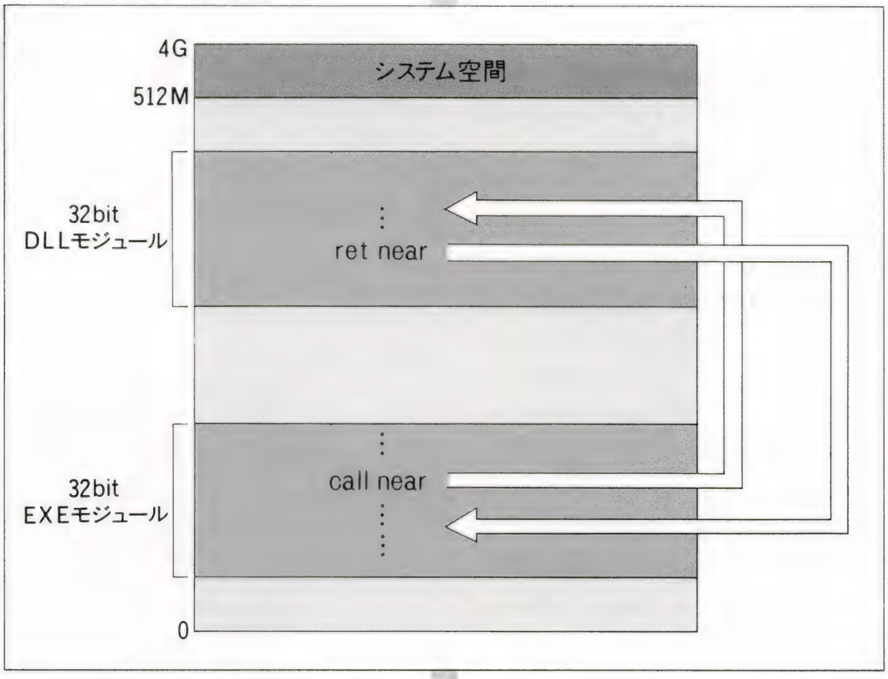
386が持つページング機能は，フラットモデルや複数のDOS互換環境を実現するために利用されている．ページングを採用したことで，メモリの管理方法もOS/2 1.xとは違ったものになっている．OS/2 1.xでは，メモリ領域が足りなくなると，セグメント単位でのスワップによって対処していた．しかし，一般にハードディスクのI／O性能は低いので，セグメント単位でのデマンドアクセス方式では，実用的な速度を得ることができない（セグメントの平均サイズは，一般にページサイズ（4Kbytes）よりも大きくなるため）．これに対し386の場合は，ページ単位でスワップを行なう．システム設計の観点からいうと，ページのほうがセグメントよりもサイズが小さいので，ハードディスクにスワップする場合のI／Oのコストは小さくなる．また，システムのメモリ管理アルゴリズムについても，セグメント単位のものより，ページ単位の場合のほうが，より効率的な方法が考案されている．このようなわけで，OS/2 2.0では仮想記憶の実現方法としてデマンドページングを採用した．

ダイナミックリンク

OS/2 2.0では，32bit DLL (Dynamic Link Libraries)形式による共有ライブラリがサポートされている．セグメントの概念がなくなった結果，DLLの概念も変化した．DLLのエントリを呼び出す場合でも，“far”の代わりに“near”を使用する．APIやDLLのルーチンを呼び出す場合でも，セグメントレジスタへのロードは必要ない．その結果，APIやDLLを呼び出すオーバーヘッドは大幅に減った．OS/2 1.xと同様，DLLは呼び出したスレッドのコンテキストで実行される．

32bit DLLは，ロード時に呼び出し側プロセスの共有メモリ空間にマップされ，

図3 フラットモデルにおけるDLLの扱い



リング3でIOPL（入出力特権）なしのモードで実行される．この条件は，OS/2 1.xのリング3 DLLとほぼ同じものだ．**図3**は，32bit DLLが，フラットモデルでどのように呼び出されるかを示している．

32bit API

OS/2 2.0は，フラットモデルのプログラムに対して32bit SE[*4] APIを提供している．32bit APIは，将来リリースされるOS/2 2.xに対して，ソースレベルでの互換性を保つことを考慮した設計になっている．

APIのアーキテクチャは，OS/2 1.xと2.0で大幅に変更されたものの1つだ．OS/2 2.0では，64Kセグメントの制限はなく，アドレスは0：32フォーマットで指定される．呼び出し方法もフラットモデルに合わせて“near call”に変更され，引数の基本長（WORDのサイズ）も4 bytes長（DWORD）になっている．全体の一貫性を保つために，一部のAPI名は変更された．また余分な機能は削除され，有用な機能が追加されている．詳しくは別稿の「OS/2 2.0が提供する32bit APIの概要」を参照していただきたい．

OS/2 1.xとの互換性

先に述べたように，OS/2 2.0では，OS/2 1.x用のアプリケーションやサブシステムをそのまま実行することができる．このために，16bit対応のプログラムと32bit対応のプログラムが，同時に実行できる環境を用意する必要がある．セグメントモデルとフラットモデルは非常に異なっているので，それぞれはまったく異なった方法で扱わなければならない．

2種類のモジュールを共存させるために必要なのは，16：16フォーマットのアドレスと，0：32フォーマットのアドレスを高速に変換するためのメカニズムである．この問題さえ解決すれば，16bit APIの処理を32bitルーチンに処理させたり，その逆を行なったりということが可能になる．セグメントモデルとフラットモデル間のアドレス変換の方法は，“LDTタイ

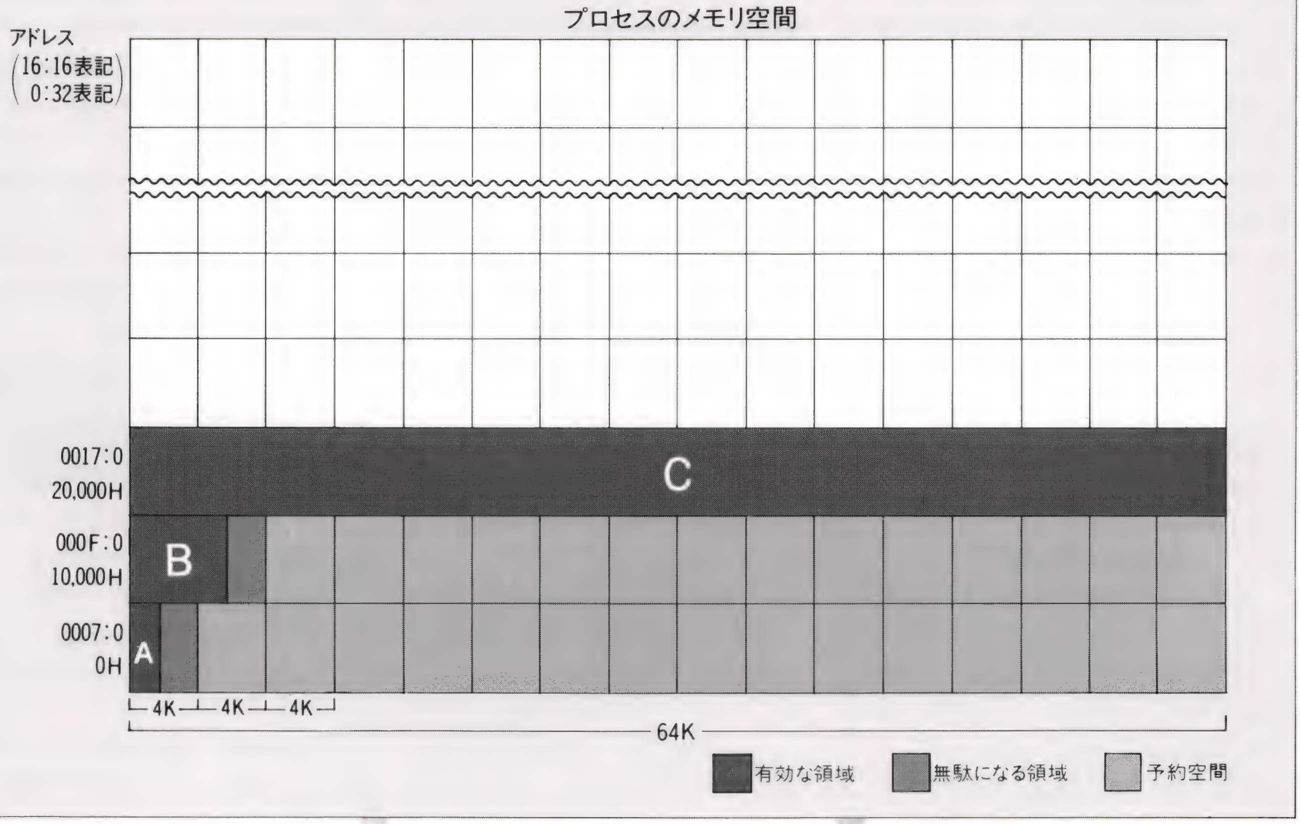
＊4 Standard Editionの略．IBMがリリースするOS/2には，基本的な機能のみを備えたSEと，データベースマネージャなどを備えたEE（Extended Edition：拡張バージョン）の2種類がある．SEは，カーネル，PM，VIOサブシステムなどのみからなる基本構成であり，Microsoftからリリースされるos/2と同等の機能を持つ．

図4 LDTタイリング

LDTの内容

セレクタ値	ベースアドレス	リミット
001FH		
0017H	200,000H	65535
000FH	100,000H	5119
0007H	0H	99

メモリオブジェクト名	A	B	C
セグメントのサイズ	100	5120	65536
ページ数	1	2	16
16:16表記のアドレス	0007:0	000F:0	0017:0
0:32表記のアドレス	0H	10,000H	20,000H

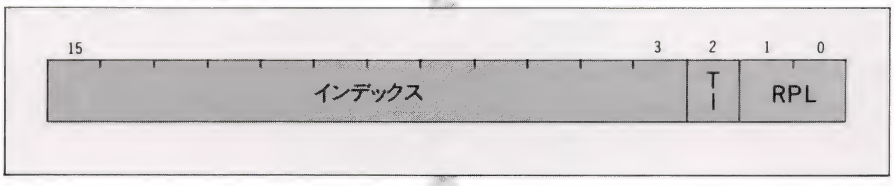


リング”(LDT Tiling)と呼ばれている.

LDTには，最大で8,192個のセグメントを登録することができる．各セグメントの最大サイズは64Kbytesである．したがって，連続するセグメントディスクリプタが，メモリ中の連続した領域を指すようにすると，1つのLDTで最大512Mbytes(64K×8,192セグメント）の連続した（フラットな）メモリ領域を管理することができる．これが，OS/2 2.0で1プロセス当たり，最大512Mbytesのメモリ領域しか利用できない理由である．

LDTタイリングを使えば，16：16モデルのアドレスと，0：32モデルのアドレスを相互に変換できる．**図4**は，16bitセグメントがメモリ中にどのように配置され，LDTにどのように登録されるかを図示したものである．16：16ポインタを0：32ポインタに変換するためには，次の式を使用する．

図5 セグメントセレクタの構造



0：32＝(SELECTOR(16：16)＞＞3)＜＜16
　　＋OFFSET(16：16)

図5に，セグメントセレクタの構造を示す．16：16モデルのセグメントアドレスから，TI(Table Indicator)ビットとRPL

(Request Privilege Level) フィールドを削除し，それを32bitアドレスの上位16bitにする．そしてオフセット部をそのまま下位16bitにすれば，0：32モデルのアドレスになる．

16：16＝MAKEP(((HIWORD(0：32))＜＜3)
+7, LOWORD(0：32))

一方0：32ポインタを16：16ポインタに変換する場合は，上位16bitを取り出し，TIビットとRPLフィールドを付加し（式中の“+7”），それをセグメントアドレスとする．下位16bitは，そのままオフセットアドレスとする．

アドレスの変換が起こるため，LDTの管理方法はOS/2 1.xとOS/2 2.0では異なったものになっている．OS/2 1.xでは，プライベートセグメントと共有セグメントは，LDT上で分散していた．それに対してOS/2 2.0では，プライベートセグメントと共有セグメントが，LDTの両端から順にアロケートされていく．16bitモジュールは，デフォルトでリロケータブルに作成されているので，この違いがあっても動作には支障をきたさない．

メモリマネージャは，16bit API経由のメモリ要求をリニアな64Kbytes境界に直し，その領域がアクセスできるようにLDTを設定する．この仕組みのおかげで，16bitのDosReallocSegで，任意のセグメントを最大サイズ（64Kbytes）まで広げられることが保証される．**図4**で，それぞれの16bitセグメントが，最低1ページの仮想／物理メモリに対応しており，ページ単位でスワップデバイスに配置されうる点に注意していただきたい．このような内部機構のため，16bitアプリケーションで小さなセグメントを数多く作成すると，ページ内部でのフラグメント[*5]が多くなる．スワップはページ単位で起こるので，フラグメンテーションが多少減ることもあるが，一般に，OS/2 1.xで同じプログラムを実行させた場合よりも，メモリの消費量は大きくなる傾向にある．これは，2つのメモリモデル間で高速なアドレス変換を行なうために，トレードオフとして犠牲にした部分である．

OS/2 1.xとの互換性を支えるもう1つの柱は，16bit APIを32bit APIに変換するレイヤである．このレイヤは，APIやパラメータの変換，APIが返すエラーコードなどを16bit API規約に合わせる働きを持っている．各APIについて，この変換を行なうルーチンを“サンク”（Thunk）と呼ぶ（厳密には，“16-to-32 Thunk”）．32bit規約のAPIを16bit規約に変換するサンクは，“32-to-16 Thunk”と呼ばれる．OS/2 2.0の初期リリースの内部構造は，まだ完全な32bit構成になっていないため，2種類のサンク両方が利用されている．サンクは，あるタイプ（16bitまたは32bit）のAPIを，もう一方のタイプに変換するためのツールにすぎない．つまり，サンクはAPIそのものではない．

複数の仮想DOSサポート

OS/2 2.0は386の仮想86機構を利用して，複数のDOS 4.0互換環境を提供している．OS/2 2.0では，DOS環境全体を仮想マシンによってカプセル化することによって，OS/2 1.xよりも互換性を高めると同時に，システムの保護も厳重にしている．OS/2 1.xでは，誤ったDOSアプリケーションが，システム全体をハングさせる危険があった．これに対しOS/2 2.0では，そのアプリケーションを含むDOSセッションがハングするだけで，ハングしたセッションはシェルから強制終了させることができる．DOSアプリケーションの実行形態としては，フルスクリーン，ウィンドウ内があり，さらにアイコン化してバックグラウンドで実行することも可能である．

厳重な保護やより高い互換性，複数のDOSセッションの同時実行に加えて，OS/2 2.0のDOSセッションにはもう1つのメリットがある．アプリケーション用のメモリが600Kbytes以上，つまり本物のDOS以上のフリーエリアがあるのだ．EMSとXMSは，両方ともサポートされている．また非同期通信ポートでは，9,600bpsまでの通信をサポートする．DOS環境もスワップの対象となるので，複数のDOSセッションを実行する場合でも，大量の物理メモリを必要とはしない．

OS/2 2.0は，DOS環境でOEMごとに特有なハードウェアをサポートする機能も持っている．これを実現するのが，仮想デバイスドライバ(VDD：Virtual Device Driver) アーキテクチャである．OS/2 2.0では，OS/2 1.xの“物理デバイスドライバ”（PDD：Physical Device Driver）が持っていたDOS用の低レベルのサポート機能がVDDに移された．386の仮想86機構では，すべての割り込みがプロテクトモードで処理されるので，デバイスドライバをバイモーダルに設計する必要はない．新しいデバイスドライバ・アーキテクチャでは，PDDがデバイスの基本的なサポートを行ない，VDDがDOS環境からのデバイスアクセスをサポートする．

デバイスドライバ

デバイスドライバの開発には，従来と同様に16bitモデルを使用する．ただし，基本的なOS/2のサポート機能と，DOS環境のサポート機能は分離されている．OS/2 1.xのデバイスドライバは，DOS環境の低レベルサポートを行なっていない限り（DevHelpのSetROMVectorを使用していない限り），OS/2 2.0でも問題なく動作する．SetROMVectorを使用しているデバイスドライバでも，OS/2 2.0で動作させることが可能だが，そのままではDOSセッションからはアクセスできない（SetROMVectorの機能が無効になっている）．また，バイモーダルに設計されたドライバも問題なく動作するが，OS/2 2.0では，常にプロテクトモード下で呼び出されることになる．

OS/2 2.0のデバイスドライバ・サポートでは，ページングに対応した拡張がな

＊5　未使用のメモリ領域が不連続に分断されること．OS/2 2.0では，ページ（4Kbytes）単位でメモリを管理するので，このようなことが起こる．

された．DMA(Direct Memory Access)を使用するデバイスは，非連続なメモリブロックへのアクセス（ページングシステムではよくあることだ）をうまく扱えないことが多い．したがって，メモリをDMA用にロックするときに，デバイスがページング環境に対応しているかどうかを，デバイスドライバで管理する必要がある．ページング環境でDMAを使用するデバイスは，“スキャッタ／ギャザー”(Scatter／Gather）という機能をサポートするようになるだろう．スキャッタ／ギャザーとは，DMAで非連続領域をアクセスするためのI/Oプロトコルである．このプロトコルをサポートしているのは，バスマスタや高機能，高速転送タイプのデバイスなどである．

デバイスヘルパ（DevHelp）には，ページロック（セグメントロックの代わりとして）と，スキャッタ／ギャザーリストの生成機能が追加された．

OS/2 2.0が提供する32bit APIの概要

Michael S. Kogan

OS/2 2.0が提供する32bit APIの仕様は，OS/2 1.xのAPIに類似したものである．とはいっても，新しいAPIではフラットモデルに従って，nearコールや0：32ポインタを使用し，64Kbytesを超えるメモリオブジェクトを扱うことができる．32bitのPM APIは，OS/2 1.xのPM APIと類似性を保っているが，OS/2 SEが提供するAPIについては，全体の一貫性などを考慮して再編成が行なわれた．OS/2 1.xが提供している機能の大部分はOS/2 2.0でも提供されるが，API名が変更されたり，機能が削除されたりしたものもある．

キーボード（KBD），ビデオ（VIO），マウス（MOU），モニタ（MON）の各サブシステムには，32bit APIが用意されていない．ただし，アプリケーションモデルとして32bit Mixedアプリケーションを利用すれば，16bit APIを呼び出すことができる（別稿の「OS/2 2.0の開発環境」を参照）．

以下は，OS/2 2.0のカーネルAPI一覧である．表中で“N／A”（Not Applicable）となっているのは，該当する機能が用意されていないことを表わす．16bit APIと32bit APIが，どのように違うかを比べてみていただきたい．

タスク管理

OS/2 2.0のタスク管理機能は，スレッドが最大4K個実行できる点を除けば，OS/2 1.xとほぼ同等である*6．スレッド管理を容易にするために，いくつかの制御機能が用意された．

DosCreateThreadは拡張され，スレッドスタック領域の動的な拡大，システムによるスタック管理，サスペンドスレッドの生成などが可能になった．新しいAPI，DosWaitThreadは，プロセス内の他のスレッドの終了を待つためのものである．DosDebugは，従来のDosPTraceに代わって，32bit対応のデバッグ機能を提供する．

OS/2 1.xのInfoSegの構造は再定義された．DosQueryInfoSegで取得可能なのは，スレッド／プロセス固有の情報を除いたものである．スレッド／プロセス固有の情報は，TIB(Thread Information Block）と呼ばれる新しい構造体に格納されており，DosGetThreadInfoによってアクセスできる．DosGetPrty，DosGetPID，DosGetPPIDなどのAPIはなくなった．これらのAPIが返す情報は，TIBに格納されている．

なお，DosR2StackRealloc，DosCallBack，DosRetForwardなどのAPIは，32bitフラットモデルでは意味のない機能なので削除されている．

＊6　OS/2 1.2では，スレッドの最大数は512であった．

OS/2 2.0 API一覧

タスク管理

16-bit Name	32-bit Name
DosCreateThread	DosCreateThread
DosCWait	DosWaitChild
N／A	DosWaitThread
DosEnterCritSec	DosEnterCritSec
DosExecPgm	DosExecPgm
DosExit	DosExit
DosExitCritSec	DosExitCritSec
DosExitList	DosExitList
DosGetInfoSeg	N／A
N／A	DosGetThreadInfo
DosGetPrty	N／A
DosKillProcess	DosKillProcess
DosSetPrty	DosSetPriority
DosGetPID	N／A
DosGetPPID	N／A
DosR2StackRealloc	N／A
DosCallBack	N／A
DosRetForward	N／A
N／A	DosDebug
DosPTrace	N／A
DosResumeThread	DosResumeThread
DosSuspendThread	DosSuspendThread

シグナル

16-bit Name	32-bit Name
DosHoldSignal	N／A
DosSetSigHandler	N／A
DosSendSignal	N／A

パイプ

DosMakePipe	DosCreatePipe

キュー

DosCreateQueue	DosCreateQueue
DosOpenQueue	DosOpenQueue
DosCloseQueue	DosCloseQueue
DosPeekQueue	DosPeekQueue
DosPurgeQueue	DosPurgeQueue
DosQueryQueue	DosQueryQueue
DosReadQueue	DosReadQueue
DosWriteQueue	DosWriteQueue

セマフォ

DosSemClear	N／A
DosSemRequest	N／A
DosSemSet	N／A
DosSemSetWait	N／A
DosSemWait	N／A
DosMuxSemWait	N／A
DosCreateSem	N／A
DosOpenSem	N／A
DosCloseSem	N／A
DosFSRamSemRequest	N／A
DosFSRamSemClear	N／A
N／A	DosCreateMutexSem
N／A	DosOpenMutexSem
N／A	DosCloseMutexSem
N／A	DosRequestMutexSem
N／A	DosReleaseMutexSem
N／A	DosQueryMutexSem
N／A	DosCreateEventSem
N／A	DosOpenEventSem
N／A	DosCloseEventSem
N／A	DosResetEventSem
N／A	DosPostEventSem
N／A	DosWaitEventSem
N／A	DosQueryEventSem
N／A	DosCreateMuxWaitSem
N／A	DosOpenMuxWaitSem
N／A	DosCloseMuxWaitSem
N／A	DosWaitMuxWaitSem
N／A	DosAddMuxWaitSem
N／A	DosDeleteMuxWaitSem
N／A	DosQueryMuxWaitSem

名前付きパイプ

DosCallNmPipe	DosCallNPipe
DosConnectNmPipe	DosConnectNPipe
DosDisConnectNmPipe	DosDisConnectNPipe
DosMakeNmPipe	DosCreateNPipe
DosPeekNmPipe	DosPeekNPipe
DosQNmPHandState	DosQueryNPHState
DosQNmPipeInfo	DosQueryNPipeInfo
DosQNmPipeSemState	DosQueryNPipeSemState
DosRawReadNmPipe	DosRawReadNPipe
DosRawWriteNmPipe	DosRawWriteNPipe
DosSetNmPHandInfo	DosSetNPHState
DosSetNmPipeSem	DosSetNPipeSem
DosTransactNmPipe	DosTransactNPipe
DosWaitNmPipe	DosWaitNPipe

タイマサービス

DosGetDateTime	DosGetDateTime
DosSetDateTime	DosSetDateTime
DosSleep	DosSleep
DosTimerAsync	DosAsyncTimer
DosTimerStart	DosStartTimer
DosTimerStop	DosStopTimer

メモリ管理

DosAllocSeg	N／A
DosAllocShrSeg	N／A
DosGetShrSeg	N／A
DosGetSeg	N／A
DosGiveSeg	N／A
DosReallocSeg	N／A
DosFreeSeg	N／A
DosAllocHuge	N／A
DosGetHugeShift	N／A
DosReallocHuge	N／A
DosCreateCSAlias	N／A
DosLockSeg	N／A
DosUnLockSeg	N／A
DosMemAvail	N／A
DosSizeSeg	N／A
DosGetResource	DosGetResource
DosSubAlloc	DosSubAlloc
DosSubFree	DosSubFree
DosSubSet	DosSubSet
N／A	DosSubUnset
N／A	DosAllocMem
N／A	DosAllocSharedMem
N／A	DosGetNamedSharedMem
N／A	DosGetSharedMem
N／A	DosGiveSharedMem
N／A	DosFreeMem
N／A	DosSetMem
N／A	DosQueryMem

DLLサポート

DosLoadModule	DosLoadModule
DosFreeModule	DosFreeModule
DosGetProcAddr	DosQueryProcAddr
DosGetModHandle	DosQueryModuleHandle
DosGetModName	DosQueryModuleName
DosQAppType	DosQueryAppType
DosGetMachineMode	N／A
BadDynLink	N／A
DosGetVersion	N／A
DosGetEnv	N／A

デバイスI/O

DosBeep	DosBeep
DosCLIAccess	N／A
DosPortAccess	N／A
DosDevConfig	DosDevConfig
DosPhysicalDisk	DosPhysicalDisk

ファイルシステム

DosBufReset	DosResetBuffer
DosChDir	DosSetCurrentDir
DosChgFilePtr	DosSetFilePtr
DosClose	DosClose
DosDelete	DosDelete
DosDevIOCTL	DosDevIOCTL
DosDupHandle	DosDupHandle
DosEditName	DosEditName
DosFileIO	DosFileIO
DosFileLocks	DosSetFileLocks
DosFindClose	DosFindClose
DosFindFirst	DosFindFirst
DosFindNext	DosFindNext
DosFindNotifyClose	DosFindNotifyClose
DosFindNotifyFirst	DosFindNotifyFirst
DosFindNotifyNext	DosFindNotifyNext
DosFSAttach	DosFSAttach
DosFSCtl	DosFSCtl
DosMkDir	DosCreateDir
DosMove	DosMove
DosNewSize	DosSetFileSize
DosOpen	DosOpen
DosQCurDir	DosQueryCurrentDir
DosQCurDisk	DosQueryCurrentDisk
DosQFHandState	DosQueryFHState
DosQFileInfo	DosQueryFileInfo
DosQFileMode	DosQueryFileMode
DosQFSAttach	DosQueryFSAttach
DosQFSInfo	DosQueryFSInfo
DosQHandType	DosQueryHType
DosQPathInfo	DosQueryPathInfo
DosQSysInfo	DosQuerySysInfo
DosQVerify	DosQueryVerify
DosRead	DosRead
DosReadAsync	N／A
DosRmDir	DosDeleteDir
DosScanEnv	DosScanEnv
DosSearchPath	DosSearchPath
DosSelectDisk	DosSetDefaultDisk
DosSetFHandState	DosSetFHState
DosSetFileInfo	DosSetFileInfo
DosSetFileMode	DosSetFileMode
DosSetFSInfo	DosSetFSInfo
DosSetMaxFH	DosSetMaxFH
DosSetPathInfo	DosSetPathInfo
DosSetVerify	DosSetVerify
DosWrite	DosWrite
DosWriteAsync	N／A

メッセージ

DosGetMessage	DosGetMessage
DosInsMessage	DosInsertMessage
DosPutMessage	DosPutMessage

コードページサポート

DosSetCp	N／A
DosSetProcCp	DosSetProcessCp
DosGetCp	DosQueryCp
DosGetCtryInfo	DosQueryCtryInfo
DosCaseMap	DosMapCase
DosGetDBCSEv	DosQueryDBCSEnv
DosGetCollate	DosQueryCollate

セッション管理

DosStartSession	DosStartSession
DosSetSession	DosSetSession
DosSelectSession	DosSelectSession
DosStopSession	DosStopSession
DosSMRegisterDD	N／A

エラー処理

DosErrClass	DosErrClass
DosError	DosError

例外処理管理

DosSetVec	N／A
N／A	DosSetExceptionHandler
N／A	DosUnsetExceptionHandler
N／A	DosRaiseException
N／A	DosUnwindException
N／A	DosSetSignalExceptionFocus

シグナル

OS/2 2.0では，シグナル関連のAPIはすべて削除された．代わりに，これらの機能は例外処理管理機構の一部として組み込まれている．Ctrl-C, Ctrl-BREAK, KILLPROCESSなどのシグナルは，例外の1つとして定義される（後述の「例外処理管理」参照）．

キュー

OS/2 2.0が提供するキューの機能は，1要素のサイズが64Kbytesを超えてもよい点と，0：32ポインタを使用する点を除けば，16bit キューAPIとほぼ同等である．ただし，混用による問題を避けるために，32bit APIのDosOpenQueueでは，16bit APIのDosCreateQueueで作成したキューをオープンできないようにしてある．

セマフォ

セマフォAPIは，他のシステムへの移植性と，OS/2 1.xとの互換性の両方に配慮しながら再設計された．RAMセマフォは他のシステムに対する移植性に問題があるため，32bit セマフォAPIではサポートされていない．

OS/2 2.0では，“private”と“shared”という2種類のセマフォを使用することができる．privateセマフォは，1プロセス当たり最大64K個まで使用できる．privateセマフォは，プロセス内のスレッドによって共有される．これに対しsharedセマフォは，全プロセスによって共有される．sharedセマフォは，システム全体で64K個まで使用できる．

これら2つのセマフォは，それをアクセスするスレッドの関係によって使い分ける．1プロセス内のスレッドのみがアクセスするセマフォには，privateセマフォを使用したほうがよい．セマフォには名前を付けることもできるが，付けなくてもかまわない．

セマフォには，privateとsharedという分類以外にも，どの系統のAPIで操作されるかによって，3種類に分類することができる．Mutex（MUTually EXclusive）セマフォはリソースアクセスの排他制御のために利用するセマフォで，MutexSem APIで操作する．Eventセマフォはイベントの通知に利用するもので，EventSem APIで操作する．MuxWaitセマフォはMutexセマフォやEventセマフォの複数待ちに利用し，MuxWaitSem APIで操作する．

メモリ管理

メモリ管理APIは，セグメントではなく，メモリオブジェクト（64Kbytesより大きくてもかまわない）を扱うようになった．OS/2 1.xのメモリ管理用APIも，フラットモデルの中で利用可能である．そして新たに，ページの属性を取得，設定するためのAPIが追加された．DosSub APIは，64Kbytesを超える領域を扱えるように修正されるとともに，中抜けヒープ[*7]（sparse heap）やスレッドの直列化制御をサポートする機能が付加された．

ダイナミックリンク

32bit APIではファミリAPI[*8]はサポートされないので，DosGetMachineModeとBadDynLinkは不要になった．また，DosGetVersionとDosGetEnvで得られる情報は，TIBに格納されているので，これらのAPIも削除されている．

ファイルシステム

いくつかのAPI名は変更されたが，ファイルシステムAPIの基本設計はOS/2 1.xと同じである．ただし，DosReadAsyncとDosWriteAsyncは削除された．これらの機能が必要であれば，アプリケーションが自力でスレッドを生成すればよい．DosReadとDosWriteは，64K超のデータを扱えるようになった．

例外処理管理

例外処理管理APIは，マシン独立性と，スレッド単位での処理（従来は，プロセス単位）を念頭に再設計された．カーネルは，16bitのDosSetVecをサポートし続けているが，32bitの例外処理管理APIでは例外処理にTIBを使用している．例外の事象は，システム例外[*9]とユーザー定義例外[*10]の2つに分類できる．16bit APIではシグナルとして扱っていた事象や，プロテクションフォールト・ハンドラの機能も，例外処理管理APIに統合された．保護ページ例外[*11]のハンドラには，動的なスタックの領域拡大ルーチンが定義されている．プロセス終了例外（エラーもしくは，明示的な終了）も追加された．C言語のライブラリで提供されるlongjmp()／setjmp()のような，ノンローカルGotoを実現するためのAPIとして，DosUnwindExceptionが用意された．DosSetSignalHandlerは，16bit APIのDosSegSigHandlerに対応している．このAPIは，システムに対して，Ctrl-CやCtrl-BREAKの例外ハンドラの登録を行なうために利用する．

＊7　確保したメモリ空間の一部にのみ，実際にアクセス可能なメモリを対応させたヒープのこと．ページ単位で設定できる，コミット／デコミットの機能を利用して実現する．

＊8　DOS環境におけるOS/2 APIのエミュレーション機能のこと．

＊9　プロテクションフォールトやページフォールトなど，通常システムが処理する例外事象．

＊10　シグナルなど，通常アプリケーションプログラムが処理する例外事象．

＊11　GUARD属性が設定されたページへのアクセスによって発生する例外．

OS/2 2.0の開発環境

Michael S. Kogan
Robert Tycast

OS/2 2.0用のプログラム開発ツールであるOS/2ツールキットは，32bit 用のCコンパイラ，マクロアセンブラ，リンカ，ライブラリ，インクルードファイルなどを含んでいる．これらのツールを使用することで，プログラマはフラットモデルや32bit APIのメリットを享受することができる．16bit OS/2環境では，プログラムの構造によって，種々さまざまなメモリモデルを管理する必要があった．これに対し32bit OS/2のプログラミング環境では，メモリモデルは1つしかない．

アプリケーションクラス

OS/2 2.0では，32bit APIと16bit APIの両方がサポートされており，プログラマは両者を混ぜて利用する(API mixing)ことができる．32bitモデルのアプリケーションから，16bit APIを呼び出したいというのは，いかにもありそうなことだ．32bitモデルを利用することで恩恵を受けられるアプリケーションは多いが，すでにリリースされているサブシステムの中には，まだ32bit APIをサポートしていないものも多いからである．OS/2 2.0のプログラムは，I/Oの使い方によって3つのクラスに，また開発工程やAPIの使い方によって4種類のモデルに分類することができる．

I/Oモデルによるクラス分けは，次のようになる．

フルスクリーン・アプリケーションクラス

フルスクリーン・アプリケーションとは，KBD（キーボード），VIO（ビデオ），MOU（マウス）などのサブシステムAPIを利用して，ユーザーI/Oを実現するタイプのプログラムである．このクラスに属するアプリケーションの中には，ビデオバッファを直接操作しているものもある．このようなアプリケーションはPMのウィンドウ内では動作しないが，大部分のフルスクリーン・アプリケーションはPMのウィンドウ内で実行させることができる．フルスクリーン・アプリケーションは，VIO APIとCの標準ライブラリの両方を使って，高速なテキストベースの処理を実現できる．

Window-Compatible アプリケーションクラス

Window-Compatibleアプリケーションには，フルスクリーン・アプリケーションの大部分（VioGetPhysBufを使用していない，つまりビデオバッファを直接操作しないもの．DosRead／DosWriteによってアクセスしているものなどである）が含まれる．printfのようなCの標準ライブラリのみで入出力を行なっているプログラムの大部分は，Window-Compatibleアプリケーションである．

このクラスのアプリケーションは，PMのウィンドウ内で実行させることができるが，PMが提供するさまざまな機能，たとえばアイコンやマウス，描画機能などを使うことはできない．

PM アプリケーションクラス

PMアプリケーションは，メッセージベースの入力キューを持つアプリケーションであり，PMが提供する豊富な機能をフルに利用することができる．

PMアプリケーションは，一貫したユーザーインターフェイスと，WYSIWYG (What You See Is What You Get)を備えている．

アプリケーションモデル

OS/2 2.0で実現可能なアプリケーションのモデルは，16bitモデルと32bitモデルに大別できる．OS/2 2.0の32bit APIは，16bit APIの機能をほぼすべて包含しているが，VIO，KBD，MOUの各コールは用意されていない．これは，将来的にはPMが，すべてのプログラムにおけるユーザーインターフェイスの基幹となるものだからである．ただし，互換性のために，VIO，KBD，MOUの各機能は16bit APIとしては残されており，OS/2 2.0の32bitアプリケーションから呼び出すことも可能である．

16bit Pureモデル

16bit用の開発ツールと，16bitリンカ(LINK.EXE)によって作成されるアプリケーションである．前述の3つのクラスすべてが，このモデルで実現できる．必要であれば，386の32bit命令を使ってもよい．しかし，32bit APIや，フラットモデルは使用できない．これは，16bit OBJフォーマットの制限である．このモデルのプログラムは，OS/2 1.xとOS/2 2.0の両方で動作するが，後述の32bit Pureモデルのように，移植性の高いプログラムを作成することはできない．**図6**は，このモデルのアプリケーションの開発フローである．

16bit Mixedモデル

このタイプのアプリケーションを作成するには，16bit／32bitそれぞれの開発ツールと，32bitリンカ（LINK386.EXE）を使用する．16bit Pureモデルと同じく，このモデルを使って，どのクラスのアプリケーションを作成することもできる．32bitリンカを使用するため，このモデルのプログラムはOS/2 2.xでしか動作しない．16bitモデルを基本にしてはいても，このモデルでは32bit命令や32bit APIを使うことができる．このモデルのプログラムは32bit EXEフォーマットに格納される

図6　16bit Pureモデルの開発フロー

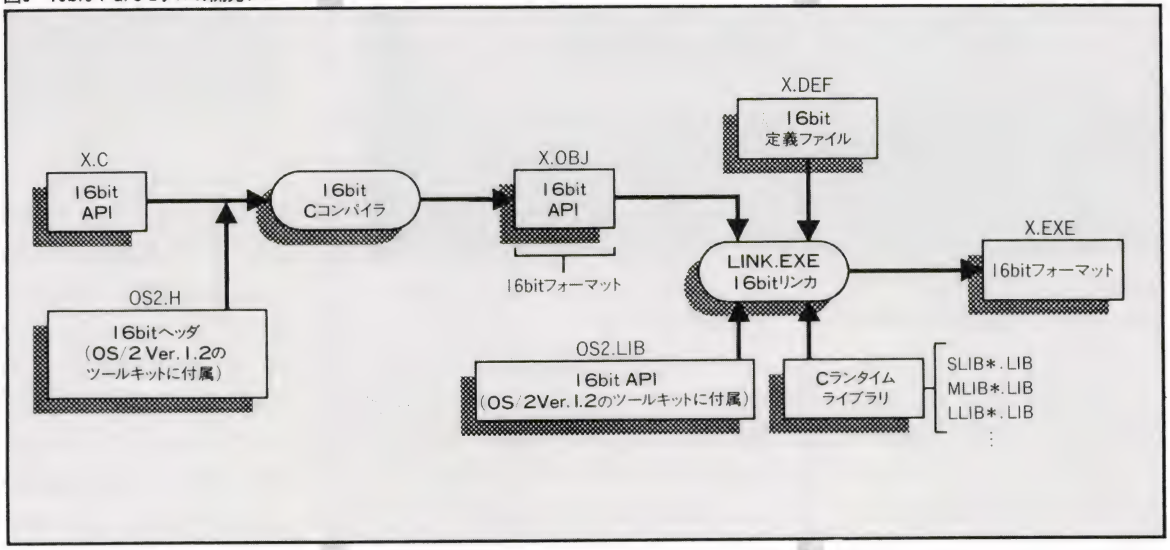
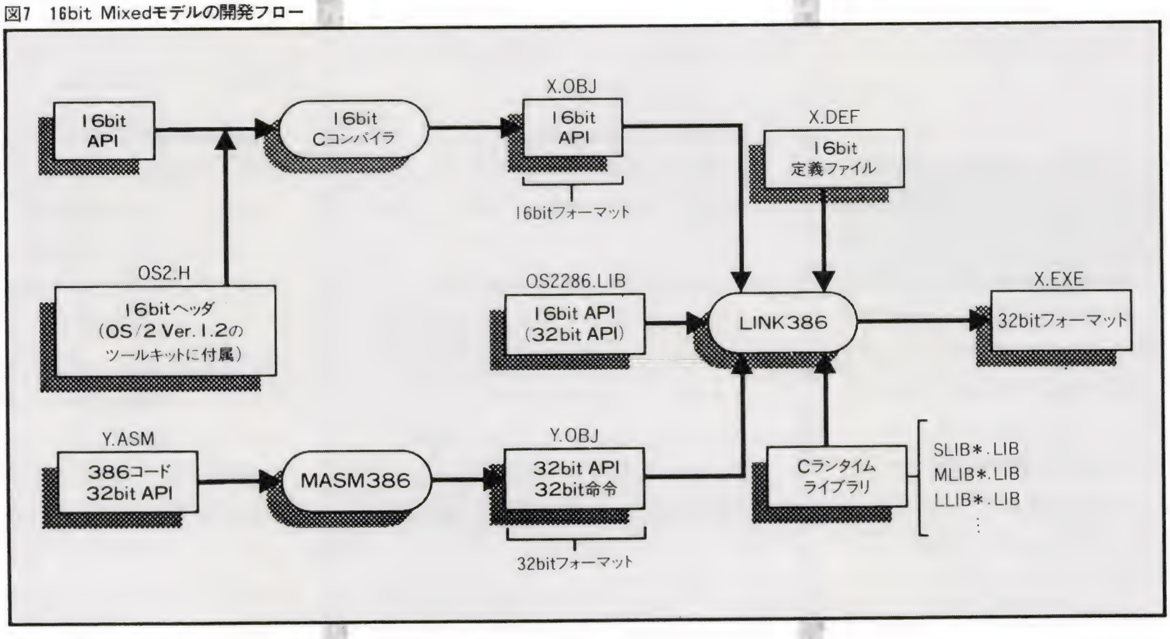


図7　16bit Mixedモデルの開発フロー



ので，OS/2 2.0のデマンドページングが利用可能だ．しかし，フラットモデルやプログラムの移植性などの恩恵を受けられないという点では，16bit Pureモデルと同様である．**図7**は，このモデルのアプリケーションの開発フローである．

32bit Pureモデル

このタイプのアプリケーションは，32bit用の開発ツールと，32bitリンカ(LINK386.EXE)を使って開発する．前述の3クラスのうち，PM／Window-Compatibleアプリケーションクラスだけでこのモデルを使用できる．

32bit Pureモデルのアプリケーションは，32bit APIだけを使用しているので，将来リリースされるOS/2のアップバージョンでも，ソース互換が保証される．VIO／KBD／MOU APIは，このモデルでは利用できないことに注意する必要がある(32bit APIが用意されていないため)．このモデルのプログラムはフラットモデルで動作し，32bit実行環境のメリットを最大限に利用できる．**図8**は，このモデルのプログラムの開発フローである．

32bit Mixedモデル

32bit Pureモデルを基本としながらも，16bit APIを使用可能にしたのが32bit Mixedモデルである．したがって，上記の3クラスのアプリケーションすべてが，このモデルのプログラムになりうる．ただし，このモデルを使用すると，32bit Pureモデルのような，アップバージョンにおけるソース互換は保証されない．**図9**は，このモデルのプログラムの開発フローである．

プログラムの移植

OS/2 2.0用のプログラムを開発する場

図8　32bit Pureモデルの開発フロー

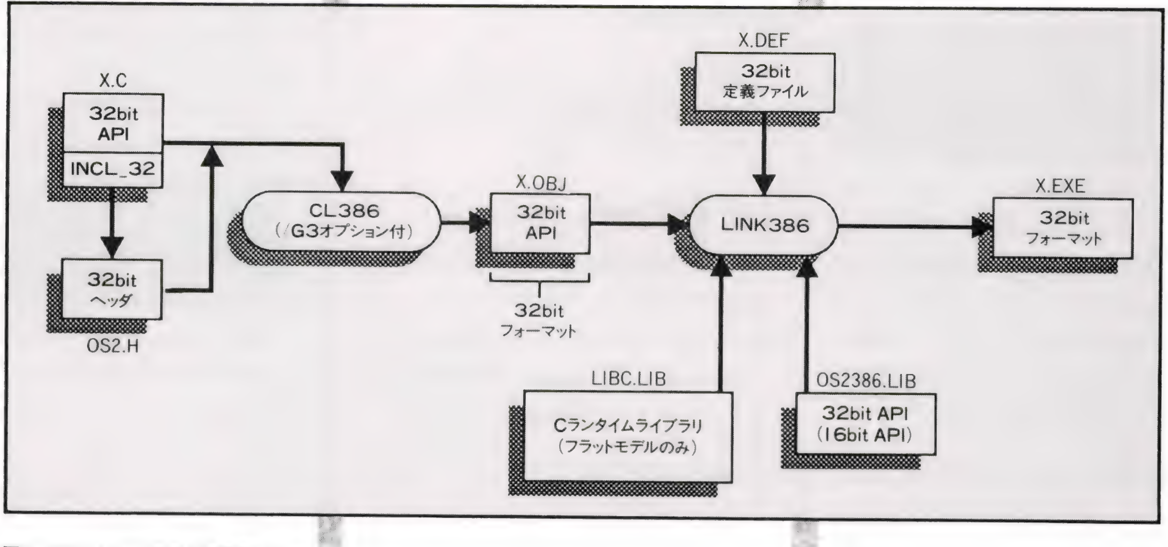
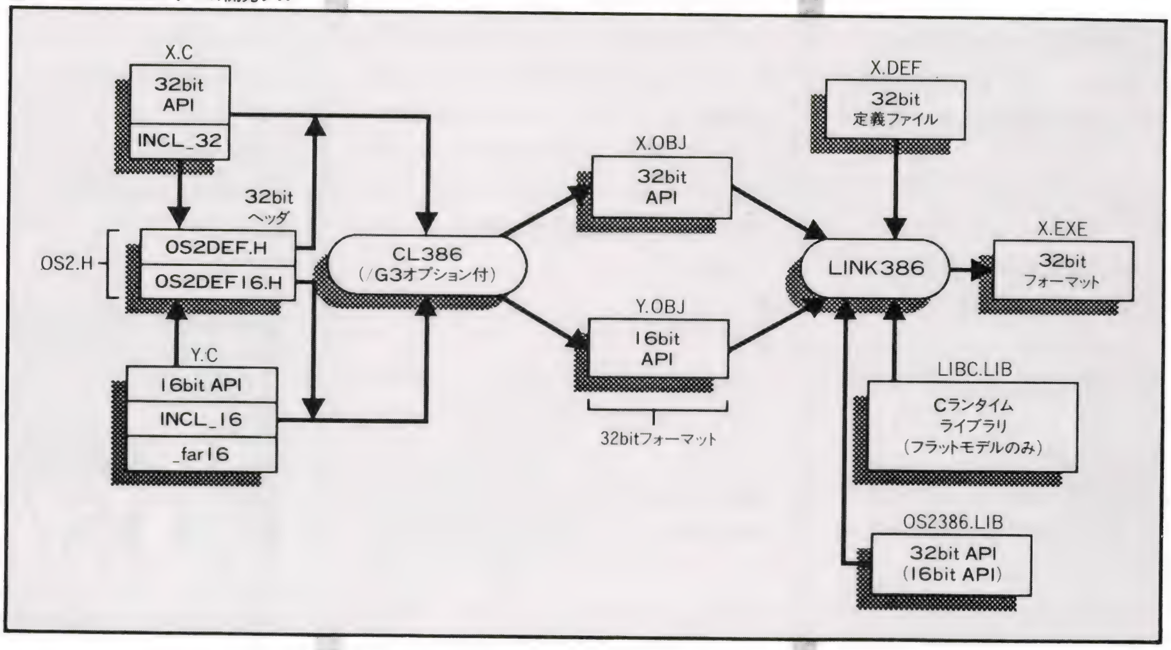


図9　32bit Mixedモデルの開発フロー



合，どのクラス，どのモデルを選ぶのがよいのだろうか．その答えは，移植しようとしているプログラムが，現在どのような環境で動作しているか，またその製品に関する短期，長期の目標がどのようなものかによって変化する．

最終的な目標になるのは，明らかに32bit PureモデルのPMアプリケーションである．しかし，PM対応でない16bitアプリケーションの場合は，２つのMixedモデルのどちらかが適しているかもしれない．16bit PureモデルのPMアプリケーションを32bit PureモデルのPMアプリケーションにするのは容易だろう．また，他のOS環境で動作しているプログラムを移植するなら，32bit Pureモデルを選択するのがよいだろう．

16bitモデルのプログラムを32bitモデルに変更するには，かなりの作業工程が必要になる．従来の16bitのCで記述されたアプリケーションでは，セグメントの利用法によってさまざまなメモリモデルが提供されており，ソースレベルでもその影響が現われている（“far”キーワードの存在など）．基本的に，これらのプログラムをフラットモデルに移植する場合には，ポインタや構造体をすべて“平坦に”する必要がある．まず，far／nearキーワードはすべて削除する．そしてCのランタイムルーチンは，セグメントモデルのものではなく，フラットモデル用のものに差し替える．OS2DEF.Hを使用すれば，32bit APIへの移行も容易なはずだ．プログラムを32bitモデルに移植するには，このような作業が必要になる．

16bitモデルと32bitモデルのAPIの違いや，メモリモデルの違いのため，再コンパイルだけで両方の環境をサポートするようなプログラムを作成することはできない．プログラム開発でどのモデルを使用するかは，最終的な目標や市場の動向，性能や移植性など，さまざまな要素を考

慮して決定しなければならない．

開発ツール

OS/2 2.0用の開発ツールは，2つのMixedモデルと，32bit Pureモデルのプログラムを開発するためのものである．16bit Pureモデルのプログラム開発には，16bit用の言語処理系やツールキットを使用する．OS/2 2.0のツールキットには，次のようなものが含まれている．

MASM386

OS/2 2.0用のMASM386は，386の32bit環境をサポートするアセンブラである．MASM386が生成するのは32bitフォーマットのOBJファイルで，これは32bitリンカ(LINK386.EXE)によって処理する．

Cコンパイラ

OS/2用のCコンパイラ(CL386)は，フラットモデルに対応している．付属するライブラリとインクルードファイルも，フラットモデル用のものである．32bit OBJファイルを生成するには，“/G3”オプションを指定する(/G3オプションは，デフォルトで指定されるようになっている)．CL386では，新しいキーワードがいくつか追加されている．“-sysentry”は，32bit API形式の関数呼び出しを指定するための修飾子である．“-far16”は，32bit環境から16bit APIを呼び出すためのもの(ポインタの変換などを行なう)であり，32bit Mixedモデルで使用される．

CodeView

32bit EXEフォーマット対応のデバッガである．基本的な機能は，16bit用のデバッガとは変わっていない．

LINK386.EXE

16bit／32bit両方のOBJフォーマットに対応した32bitリンカである．ただし，生成できるEXEフォーマットは32bit EXEのみであり，DOSや16bit OS/2のEXEフォーマットはサポートしていない．

システムライブラリ

16bitのツールキットに含まれていたシステムライブラリは，OS2.LIBだけであった．それに対して，32bit OS/2ツールキットには，OS2286.LIBと，OS2386.LIBの2種類のシステムライブラリが用意されている．OS2286.LIBは16bit Mixedモデル用のライブラリであり，OS2386.LIBは32bit Pure／Mixedモデル両用のライブラリである．

インクルードファイル

OS2.Hは拡張され，“INCL_32”を指定すると32bit API用の宣言が取り込まれるようになっている．一方“INCL_16”を指定すると，32bitモジュールから16bit APIを呼び出すための宣言が取り込まれる．1つのモジュールで両方の宣言を読み込むことはできない．ただし，異なるタイプのAPIを呼び出すモジュールをリンクすることは許されている．

/G3オプションを指定してコンパイルすると，デフォルトでINCL_32が指定される．/G3とINCL_16を組み合わせた場合には，16bit APIを使用する32bit モジュールが生成される．またコンパイル時に/G2オプション[*12]を指定すると，OS/2 1.x 互換のAPIの宣言が取り込まれる．

終わりに

OS/2 2.0のプログラミング環境では，プログラムの開発や32bit環境への移植を促進するために，さまざまなアプリケーションクラスとアプリケーションモデルがサポートされている．32bitプログラミングモデルは，16bitモデルに比べると単純だが，16bitと32bitのアプリケーション双方のAPIを単一システムでサポートしたことは，複雑化の原因になっているといえる．しかし，OS/2 2.0が提供する環境を適切に利用すれば，386ベースのシステムの機能をフルに引き出し，移植性にも優れたアプリケーションを開発できるだろう．

＊12　286用のコードを生成するための指定．

筆　者　紹　介

Michael S. Kogan

現在フロリダ州のNova Universityにてコンピュータサイエンスの博士号取得のため離職中．元IBM OS/2 Design Staffのプログラマ．

1984年，IBMに入社，ボカラトンにてIBMのエンジニアリング，サイエンス関連の製品開発に従事．その後，IBM PC用のXENIXの開発とインプリメントを行なった．16bit OS/2の開発に2年，32bit OS/2の設計にも2年半参加している．数学とコンピュータサイエンスの学士号(Emony University in Atlanta, Georgia)と，コンピュータサイエンスの修士号（Nova University）を持つ．

Robert Tycast

OS/2デザイン部門の顧問プログラマ（advisary programmer）．現在，OS/2 2.0のインフォメーション・プログラミングツールのプロジェクトに従事．1989年に，Digital Equipment Corporation(DEC)からIBMに入社．DEC在職中は，米国，ラテンアメリカ，ヨーロッパなどでソフトウェア開発とサポートに従事した．Xウィンドウ（Ver.11），AI技術（LISPやOPS5），テクニカルワークステーション（VMSやULTRIX）などのプロジェクトを経験している．Massachusetts Institute of Technology（MIT）の学士号を持つ．

TEST LAB.

テストラボ

今回紹介するTool Bookは，MS-Windows Ver.3.0上で作動する，いわゆるハイパーテキストだ．Macintoshの Hyper Cardに近く，そのHyper Cardのスタックをコンバートするソフトなども発売がアナウンスされている．現在，最も注目すべきソフトである．SPARCstation SLCは，ディスクレスとはいうものの100万円を切った価格に魅力がある．今後の他社の動向にも少なからず影響を与える“マイルストーン”ともなるマシンだ．今回は，ディスクレスマシンの使い方もあわせて解説する．

ToolBook
ハイパーテキストで綴る本の世界

SPARCstation SLC
ディスクレスで低価格それでいて高性能

QMS ColorScript 100 Model 10J
日本語PostScript搭載のカラープリンタ

ToolBook

ハイパーテキストで綴る本の世界

MS-Windows Ver.3.0（以下，Windows3.0）は，5月22日に発表されたMS-Windowsの最新バージョンである．80386のネイティブモード（386 Enhanced mode）で動作することや，従来のVer.2.11に比べ，ユーザーインターフェイス（Look＆Feel）が劇的な変化を遂げていることなどから，最近，何かと取り上げられる機会が多い．ここでは，そのWindows3.0にランタイムルーチンが搭載されたことで，当初から話題になっているAsymetrix社のToolBookを取り上げる．

ToolBookはWindows版HyperCard

ToolBookは，MacintoshのHyperCardに見られるようなハイパーテキスト環境を，Windows上で実現するソフトウェアである．

ただ，HyperCardとは異なり，ToolBookが実現するハイパーテキスト環境は，あたかもそれがパソコン上に存在する1冊の「本（Book）」であるかのように見える点が特徴になっている．つまり，「本」がHyperCardの「スタック」に，「ページ」が「カード」に対応する．

その各ページには，普段，私たちが目にする本と同様に，テキスト（文章）データや，挿絵，図表，写真などのグラフィックデータはもちろん，アニメーションや音楽データなどのさまざまな情報を盛り込むことが可能になっている．

加えて，他のページや任意の情報をリンクするためのボタンやアイコンなどの部品（オブジェクト）が，そのページ上に配置される形式となる．

また，文章中に任意の単語（Hotwords）に対して，注釈や別の本との参照関係を作ることも可能である．

Quick TourでToolBookの旅

ToolBookのインストールには約7Mbytesのディスク容量が必要である．インストールが完了するとWindows3.0のProgram Managerウィンドウに，「ToolBook」と名付けられたサブウィンドウ（プログラムグループ）が開かれる（**画面1**）．

その新たに作成されたプログラムグループには，「Quick Tour」，「DayBook」，「ToolBook」という3冊の本が収められている．

「Quick Tour」は，ToolBook自体の解説書になっていて，これを読めば，とりあえず，ToolBookがどういうものなのか，その概要を理解できるようになっている．

このQuick Tour自体も，ToolBookの1つの本として作成されていて，その解説を読みながら，ハイパーテキストによる本を即座に体験できるようになっているのだ．

画面2は，そのQuick Tourを初めて起動したときの冒頭のページを示したものである．ここで，Yを選択すると，マウス操作の解説が現われる．それらのページを見ながら，指示どおり操作すれば，この章（トピック）を読み終えたときには，ToolBookにおけるマウス操作の大体のコツがマスターできるという寸法だ．

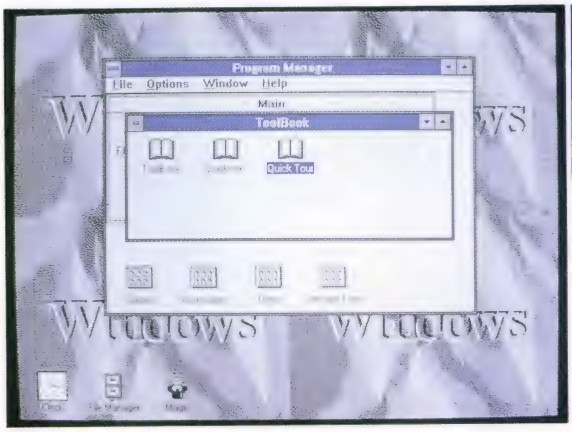
このマウス操作編を読み終えると，次に，“Quick Tour Topics”とタイトルされたページが現われる．そのページの右側には，“Using the Mouse”から始まって，“The ToolBook Auther Tour”で終わる，Quick Tourの各トピックが示されていて，その全体の構成をひと目で知ることができる．このページには，この本を縦横無尽に旅するためのさまざまな仕掛けも盛り込まれている（実は，ここからがハイパーテキストの本領発揮といったところなのだ）．

たとえば，すでに読み終えた“Using the Mouse”の左側には，それを示すマークが記されている．

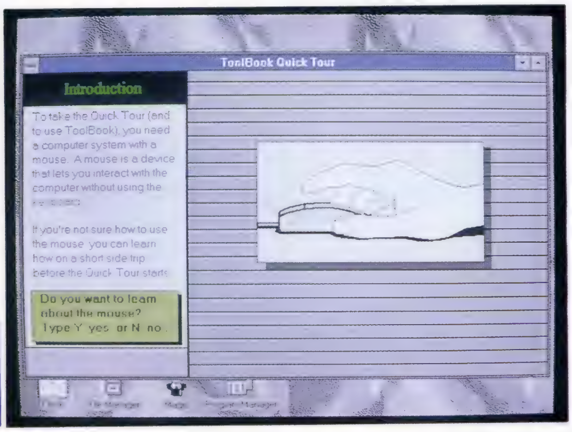
それぞれのトピックの上にマウスポインタを移動すると，そこに書かれている事柄の簡単な要約がポップアップ表示される（**画面3**）．

また，ページ左下の“Map”というフィールドをクリックすると，“Quick Tour Map”というページが開かれ，この本の各トピックの階層構造を視覚的に見ることも可能だ（**画面4**）．

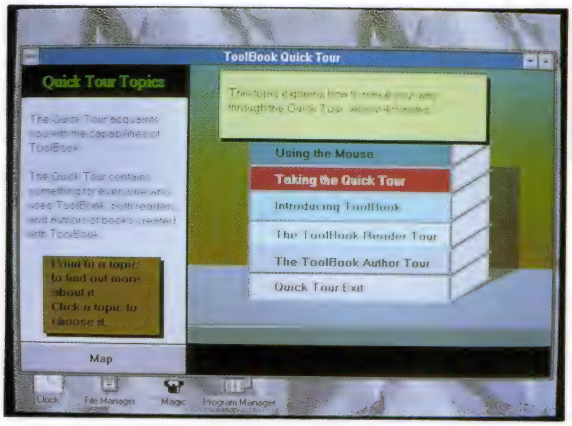
それぞれの各トピックの内容を読み進



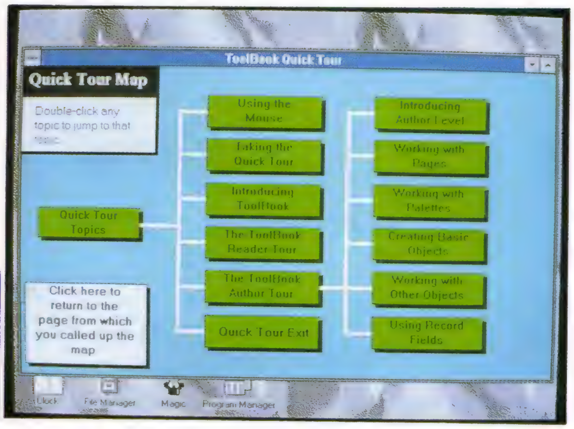
画面1　Windows上のProgram Managerで，ToolBookのサブウィンドウを開いた．Quick Tour，DayBook，ToolBookの3つの本型のアイコンがある．



画面2　Quick Tourを起動するとマウスの絵が描かれている画面になる．



画面 3　“Quick Tour Topics”のトピック上にマウスポインタを移動すると，その要約がポップアップ表示される．おもしろい機能だ．



画面 4　“Quick Tour Map”の中の“The Tool Book Auther Tour”をクリックすると，その下の階層のトピックが表示される．

むときは，該当のトピックをマウスでクリックするだけだ．たとえば，上から2つ目の“Taking the Quick Tour”をクリックすると，このQuick Tour自体の使用方法の説明の最初のページが開かれるといった具合である．

先の“Quick Tour Map”のページを利用すれば，より階層の深いトピックへ直接ジャンプすることも可能だ．

このように，ハイパーテキスト環境ならではの操作性を，Quick Tourでは堪能できる．ちなみに，このQuick Tourをすべて読破するには，約2時間ほどの時間を要する．と，Quick Tourの最初のページには書かれている．

DayBookは，スケジュール&アドレス帳

2つ目のアイコンである「DayBook」は，万年カレンダーを内蔵したスケジュール管理機能を実現するサンプルアプリケーションである．そして，このDayBookという本は，あたかも，Windowsのアプリケーションプログラムであるかのような振る舞いをする（変な表現だが，事実だから仕方がない）．月間，週間，日程表の各スケジュールを記述可能なカレンダー機能に加えて，アドレス帳などの機能がある（**画面5**）．

このサンプルを見ていると，なかなか実用的なアプリケーションに仕上がっていて，ToolBookが単純な読み出し専用の本を作成するためのツールではなく，さまざまなビジネスアプリケーションが作成可能なことを実感させてくれる．

本の整理は本棚（Bookshelf）で……

そして，最後のアイコンである「ToolBook」を選択すると，「Bookshelf」という，書棚をイメージするウィンドウが開く（**画面6**）．そのウィンドウ中には，1冊の本が見開きの状態で描かれ，その見開き一面に本の形をしたアイコンが並んでいる．本の下には，そのタイトルが書かれている．このBookshelfが，“本を格納するための書棚”の役割を果たしており，ToolBookのメインウィンドウといった雰囲気になっている．ちょうど，HyperCardのHomeStackに当たる．

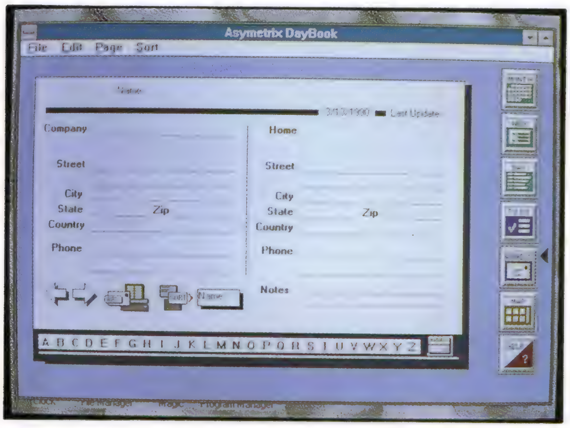
この「書棚」には，当初から多くの「サンプル本」が含まれている．たとえば，Applicationsに分類されている「Calculator」という本は，電卓プログラムそのものである．先のQuick TourやDayBookは，ここにも登録されている．

「Page Ideas」や「Clip Art」などのように，新たにオリジナルの本を作成するときに，そのまま利用できるページのフォーマット（台紙）や，ページに配置するボタン類やアクセサリ見本などを数多く収めた本もある（**画面7**）．

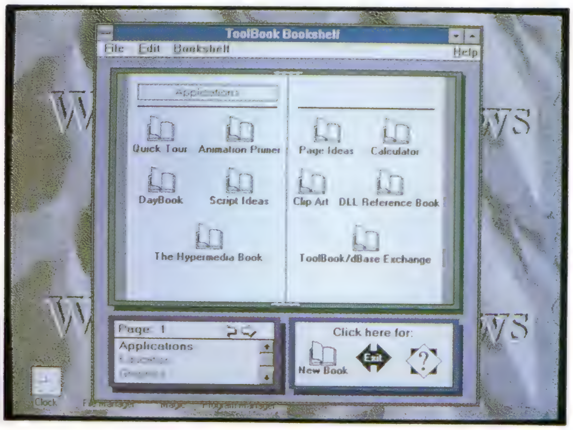
また，「Animation Primer」のように，アニメーションを実演してくれるものもあったりする．

メモ帳を作る

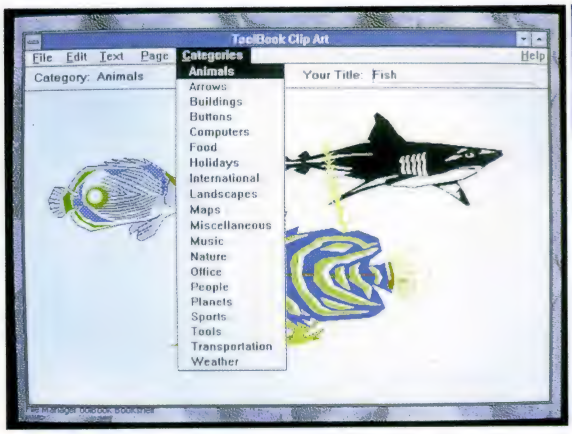
ToolBookの概観が分かったところで，試しに，簡単な本を作ってみた．ここでは，その新たに作成した本を見ながら，その内部構造をのぞいてみる．



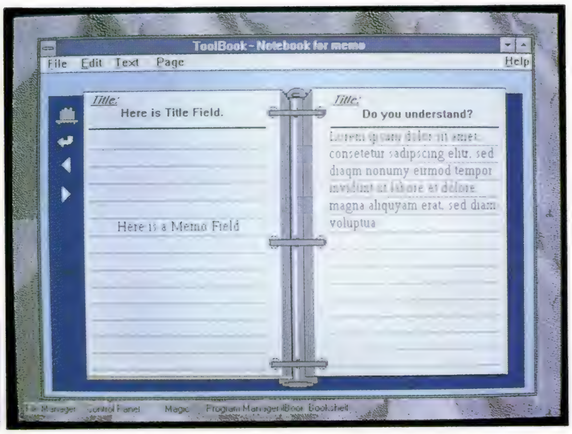
画面 5　DayBookのアドレス帳．これを参考に自分なりのアドレス帳を作ることもできる．



画面 6　Bookshelfのウィンドウ．Page IdeasやClip Artなどの本が並んでいる．



画面 7　Clip Artは，オリジナルの本を作るときにそのまま利用できる絵などを収めた本だ．



画面 8　memo.tbkの起動画面．Page Ideasよりフォーマットを選択して作成した．

作成したのは，単純なメモ帳で，システム手帳をイメージするデザインを選んだ（**画面 8**）．その用紙のフォーマットは，タイトルを記入する部分と，本文を記述する２つの部分からなっている．この用紙の任意の位置にカーソルを移動し，キー入力すれば，メモを自由に書くことができる．

左側の裏表紙には，ページを繰るためのアイコンが４つあり，上から順に，“Bookshelfの呼び出し”，“そのページを開いた直前のページへ戻る”，“次のページへ移動”，“前のページへ移動”の各機能を果たす．

この手帳に，新たなページを追加するときには，Pageメニューの「New Page」コマンドを実行する（**画面 9**）．

本をデザインする

新しい本を作成するには，単に，Bookshelfウィンドウの「New Book」アイコンをクリックすればよい．すると，真っ白なキャンパスのような，何も書かれていない状態の本が開く（**画面10**）．

そして，本の作成時や編集中のウィンドウの左上には，必要なアイコンなどのオブジェクトを，ページ上に配置するために使用するパレットが置かれる．

このパレットを用いて，ページの図柄や，アイコンなどを描いていけば，オリジナルの本が作成できるわけだ．また，図柄やイラストなどは，Windows3.0に付属するペイントツール「Paintbrush」で作成したグラフィックデータを取り込むことも可能だ．

ただ，この方法だと，ゼロからそのすべてを記述しなければならないので，初めてでは少し荷が重い．そこで，先に紹介したBookshelfの中の「Page Ideas」の本の中から，適当なものを切り出して使用することにした．それを，再編集してできあがったのが，**画面 8** のメモ帳だったわけである．

アイコンの動作を定義する

実は，こうして切り出した本（手帳）には，先に紹介した４つのアイコンが，すでに組み込まれた状態になっている．

ToolBookでは，これらのアイコンの動作を，“OpenScript”と呼ばれるオブジェクト指向型のプログラミング言語を使用して記述する仕組みになっている（HyperCardのHyperTalkのようなものだ）．つまり，各アイコンの上に，それぞれの動作を記述したスクリプトを書き込んでいくわけである．

では実際に，アイコン上に記述されたスクリプトのようすを見てみよう．

そのためにはまず，「Page」メニューの「Background」コマンドを実行して，編集の対象となるページを，アイコンが設定されているバックグラウンドページに切り替える．

その後，参照するアイコンをマウスで選択し，さらに，「Object」メニューの「Button Properties...」コマンドを実行すると，そのアイコンの属性などを設定するためのダイアログボックスが現われる（**画面11**）．ここで，「Script」ボタンをクリックすれば，Scriptウィンドウが開いて，目的のアイコン上に設定されているスクリプトが表示される（**画面12**）．あとは，このScriptウィンドウを使用してスクリプトを編集すればよい．

すなわち，このOpen Scriptは，HyperCardのそれと同様に，各アイコンごとに設定された「紙（ウィンドウ）」の上にプログラミングしていく仕組みになっているのだ．

Scriptウィンドウからも読み取れるように，このアイコンの動作は，２つのメッセージハンドラ（全部でたったの６行）によって定義されている．最初の３行，

```
1: to handle buttonup
2:     send next
3: end buttonup
```

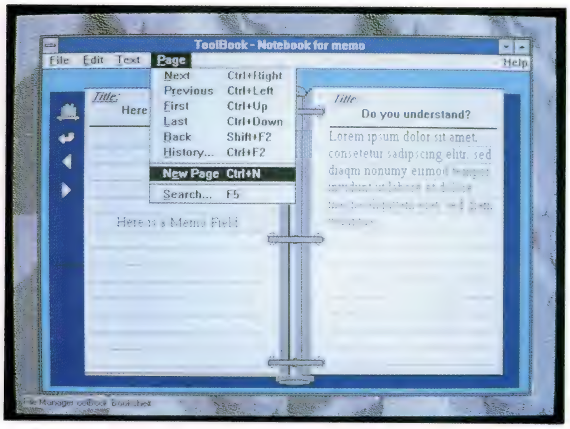
というスクリプトは，次のページへ移動するアイコンの動作を実現するものだ．

１行目のto handle buttonupという構文は，このメッセージハンドラの始まりを示すもので，このアイコンが，“マウスのボタンが押し離されたというメッセージ（＝buttonup）”を受け取ったとき，このスクリプトの実行を開始することを意味している．もし，このスクリプトをボタンが押された瞬間に実行したいなら，この“buttonup”の代わりに，“button down”と書けばよい．

２行目のsend nextは，“next”というメッセージをシステムに送ることを意味している．このステートメントの実行によって，実際の改ページ動作が行なわれる．

そして，３行目の end buttonup で，このスクリプトは定義を終了する．

このように，アイコン上で行なわれた操作は，逐一，メッセージ（buttonupや



画面 9　New Pageコマンドにより，新しいページを追加することができる．

画面10　1から本を作成するときには，New Bookを使う．ただ，New Bookではじめから作る作業はたいへんである．

buttondownなど）に変換されて，その動作を記述した，該当のハンドラ（Open Script）が起動される仕組みになっているわけである．

たとえば，さらに，このアイコン上で，マウスボタンが押された瞬間に（離される前に），ブザーを鳴らすには，以下のようなスクリプトを，上記のものに加えればよいわけだ．

```
1: to handle buttondown
2:      beep 1
3: end buttondown
```

リスト 1～3 は，メモ帳のそれぞれのアイコンのOpenScriptを示したものである．基本的には，2行目で送信するメッセージが異なっているだけだ．

こうして，スクリプトの編集を終えると，あとは，そのScriptウィンドウを閉じる．すると，そのアイコン上に，編集したスクリプトがコンパイル処理されて自動的に登録される．このとき同時に，その構文検査も行なわれ，もしエラーがあれば報告される．

本の動作を定義する

ToolBookでは，上記のように，アイコンごとに，その動作を記述するほかに，スクリプトの階層だった記述が可能となっている．

たとえば，**リスト 3** のスクリプトは，bookshelfというメッセージをシステムに送信するものだが，ただ，そのメッセージを処理するbookshelfというメッセージハンドラ（スクリプト）は，どこにも定義されていない（ほかのハンドラが送信するnext，previous，backは，すべて，ToolBookのランタイムライブラリにあらかじめ定義されている）．

実は，このbookshelfというメッセージを処理するハンドラは，アイコンが配置された土台（台紙）となっている本そのものに定義されているのだ．

実際に見てみよう．同じ「Object」メニューから，今度は「Book Properties...」コマンドを選択し，Scriptウィンドウを開ける．**リスト 4** は，そこに定義されている全スクリプトを示したものである．16行目に問題のスクリプト（Book shelfハンドラ）が定義されている．

つまり，このようにして，**リスト 3** のハンドラによって送られたbookshelfメッセージは，本自体の上に記述されているさらに下位の階層の“to handle book shelf”で始まるメッセージハンドラに送られて処理される．

このように，ToolBookでは，スクリプトを階層化することによって，アイコンなど，ページに配置される複数のオブジェクトに共通する動作を，あたかもサブルーチンのように記述することが可能になっている．

また，この本の階層では，本がオープンされたときに自動的に実行される処理手続き（**リスト 4**，**1 行目**）や，その逆に，本が閉じられるときに実行される手続きなども記述できるようになっている．

豊富なランタイムライブラリ

当然，このOpenScriptでは，C言語などのプログラミング言語と同様に，さまざまなタイプの変数や制御構造（If／then／else，While，Do／untilなど）を柔軟に記述することが可能だ．

そういう意味では，このToolBookをウィンドウプログラミング用の独立した言

リスト 1　1つ前のページに移動するスクリプト

```
1: to handle buttonup
2:        send previous
3: end buttonup
```

リスト 2　オープンした順にページを1つずつさかのぼるためのスクリプト

```
1: to handle buttonup
2:        send back
3: end buttonup
```

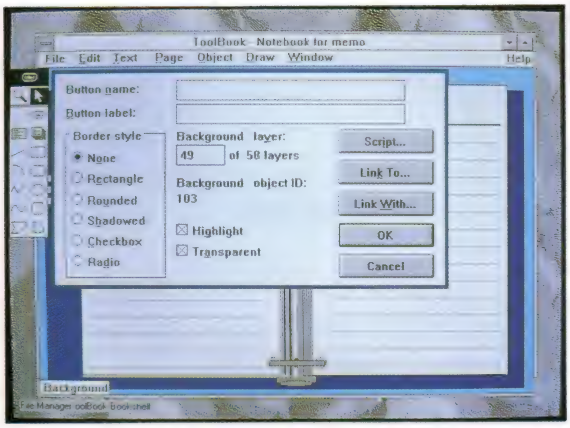
リスト 3　Bookshelf（本棚）を呼び出すためのスクリプト

```
1: to handle buttonup
2:        send bookshelf
3: end buttonup
```

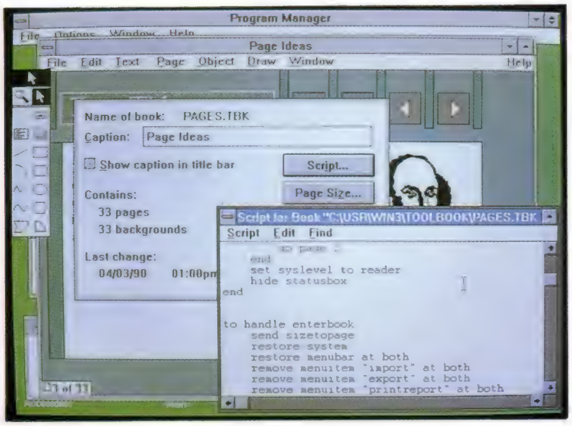
リスト 4　本のスクリプト（一部）．

```
1: to handle enterbook
2:    send sizetopage
3:       restore system
4:       restore menubar at both
5:       linkdll "user"
6:    int bringWindowToTop(word)
7:     end
8: end
```

（後略）



画面11　ここで，指定したアイコンの属性を設定する．



画面12　スクリプトウィンドウ．ウィンドウに表示されているのは，このページのスクリプトである．

語処理系としてとらえることもできる．

その処理系の機能を裏で支えているのが，“Windows Dynamic Link Libraries”と呼ばれるランタイムライブラリ（“DLL”という拡張子を持ったファイル）である．この中には，単にページを移動するといった基本的な動作を制御するものだけでなく，ウィンドウのメニューバーを編集したり，ウィンドウのサイズを変更するなどのWindows自体を操作するためのファンクションも数多く収められていて，OpenScriptをより強力な言語にしている．

このランタイムライブラリの呼び出しは，それがあたかも最も下位の階層に定義されたスクリプトであるかのように見える．

すなわち，あるアイコン上で定義されたハンドラが（send文などを使用して）送信したメッセージは，まず，そのアイコン上に定義されている同名のメッセージハンドラによって処理される．もし，そこに定義されていなければ，1つ下の階層（たとえば，そのアイコンが置かれているページ）の上に定義されたハンドラが実行される．そこにもなければ，その下の本の上に定義されたものが実行されるといった具合に，そのメッセージは順次，下の階層に伝えられ，すべての階層になければ，最後に，ライブラリに渡

ToolBookとHyperCard

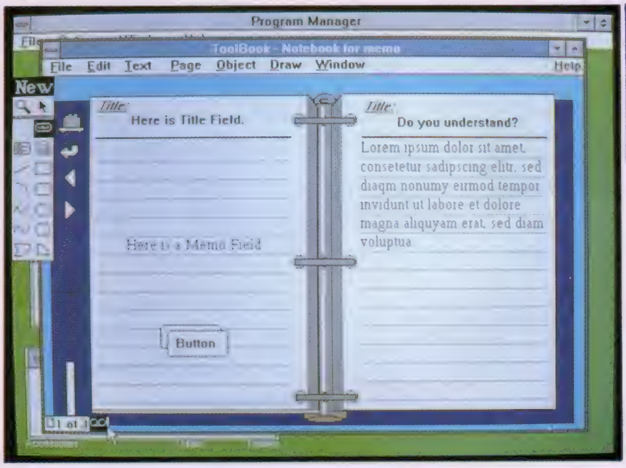
ToolBookはその基本コンセプト，外見ともにApple社のHyperCardから多くを学んだことに間違いはない．HyperCardを触った経験があれば，ToolBookの基本的な操作に戸惑うことはないというほど，ソックリに仕上がっている．

それはメニューの構成にまで見られるほどなのだ．ToolBookに搭載されたスクリプト言語「OpenScript」は，ボタン，ノート（カード），ブック（スタック）と階層をなし，それぞれのオブジェクトに収められる点でもHyperCardの「HyperTalk」と同じ構造である．少しでも腕に覚えがあれば，HyperTalkのスクリプトをOpenScriptに書き換えることも，それほど難しくないだろう．各オブジェクトのプロパティ（属性）もHyperCardに準ずる構成で，ほとんどそのままといっても過言ではない．各プロパティの値をOpenScriptで制御できる点もHyperCardと同様だ．

とはいっても，ToolBookは，スクリプトに関連してHyperCardにないユニークな機能も搭載している．ユーザーが操作したマウスの動きやキーストロークを記録し，スクリプトを自動的に作成してくれるScriptRecorderがそれだ（**画面A**）．このScriptRecorderを使えば，ページ上で新しいボタンを作成するなどのアクションやオートデモのような動作を簡単にスクリプトにすることができる．

もう1つの違いとして，ToolBookにはスクリプトコンパイル機能が搭載されていることが挙げられる．この機能は，スクリプトウィンドウで編集後，スクリプトを逐次コンパイルしながら各オブジェクトに収納するものだ．また，併せて搭載されているデバッグ機能で，スクリプティング作業中にシンタックスのチェックを行なうこともできる．

それに対して，HyperCardは先日発表されたVer.2.0で，一度実行したスクリプトをメモリ中にキャッシュする機能を備えたが，基本的にインタープリタ形式である．ちなみにHyperCard用のスクリプトコンパイラは，サードパーティからリリースされている（**画面B**）．



画面A　ToolBookのScriptRecorder機能では，ウィンドウ左下端にカセットテープのアイコンが現われてスクリプトの記録動作中であることを示す．

されるといった仕組みである.

OpenScriptを速やかに記述する機能

このように，本の中には，さまざまな階層にOpenScriptがちりばめられて，各オブジェクトの動作は実現されていく．当然，複雑なハイパーテキスト環境を実現するには，それ相応のスクリプトが，その裏側に隠れているはずである．その一部始終を記述するのはたいへんなことである．特にアニメーションなどの動作を記述するのは想像しただけでも骨が折れる作業であることに違いない．

そのため，ToolBookにはオブジェクトに対するマウスの操作などを，そのままスクリプトに翻訳する機能や，スクリプトの動作を確認するためのソースコードによるデバッグ機能など，新たな本を作成するためのサービス機能が数多くサポートされている．

また，「Script Ideas」から切り出したり，Bookshelfに収められている多くの本の，実際のスクリプトを参考にするのもよい．

ちなみに，今回のメモ帳も，ほとんど，そのようにして作成したもので，作業を開始してから完成するまで30分と要さなかった．

ToolBookの可能性

これまで見てきたように，ToolBookは，あくまでもハイパーテキスト環境を実現するプログラミングのための道具である．そういう意味では，ワードプロセッサやスプレッドシートなどのアプリケーションソフトのように，それを購入したからといって，即座に役立つものではない．

しかし，ウィンドウ環境でのプログラミングの難しさがいわれ始めた昨今，このToolBookの十分に強力なOpen Scriptを使用して，はるかに容易に，短時間で，自分なりのWindowsアプリケーションが作成できるのは，非常に魅力的なことである．それにも増して，Windows上で，本格的なハイパーテキスト環境が実現されたことは，たいへん有意義なことではないだろうか．

このToolBookによって，自分なりの情報整理学を構築してみるのはどうだろう．さらなる，パソコンの有意義な活用法が見つかるかもしれない．

ただ，80286の10MHzでも動作可能ではあるが，速度が遅く快適とはいえなかった．できれば，80386SXの16MHz以上の環境をお勧めしたい．

（日進ソフトウエア㈱　矢部和博）

製品名	ToolBook
対応	MS-Windows Ver.3.0 以上
価格	395ドル
問い合わせ先	ジィスクソフトウェア㈱ Tel:03-400-3121

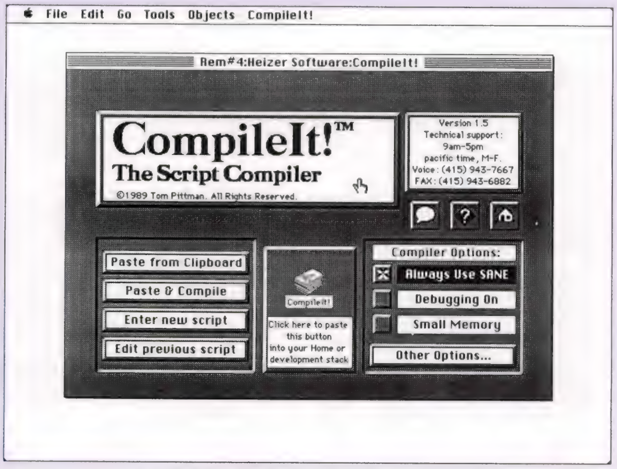
ToolBookの成功の条件

HyperCardは1987年の夏に登場して以来，たいへんな成功を収めたソフトウェアプロダクツである．では，果たしてToolBookもHyperCardに匹敵する成功を収めることができるのだろうか．それには，いくつかのポイントがあると思う．ToolBookの成功を占う意味で，HyperCardの成功の要因を少しまとめてみた．

HyperCardの成功の大きな要因として，Macintosh本体にバンドルされたことが挙げられる．これにより，ほとんどすべてのMacintoshユーザーが時期をおかずに同一のソフトウェアプラットフォームを得たことになる．この意味は重大で，HyperCardがエンドユーザーにとって馴染みやすいアプリケーション作成ツールであるという以上に，サードパーティにとってもHyperCard上のアプリケーション（スタックウェア）やHyperCard関連の各種ユーティリティを開発・販売するに足る市場が現われたことを意味するからだ．その相乗効果により，さらにローエンドのユーザーがHyperCardのスタックウェアの作成に取り掛かるときには，十分なリソース（再利用可能なスクリプトを収めたボタンなど）が揃えられることになった．

以上のような観点からすると，ToolBookの成功は，MS-Windows Ver.3.0（以下，Win 3）の普及と一蓮托生であるといえる．ToolBookが1つの市場をなすようになるまでには，まずWin 3が十分に行き渡らなくてはならない．そのうえ，早い時期からToolBookに興味を示し，しかも多くのブックを作成してくれるユーザーが必要だ．Heizer Software社よりHyperCardのスタックをToolBookにコンバートできる「ConvertIt！」といったソフトウェアも登場している．このように，ToolBookではHyperCardの多くの資産を流用することができる．この点は，普及の1つの条件である．十分なリソースの供給を満たすことになる．ただ，Win 3にはToolBookのプレイヤー（ランタイムライブラリ）が付属するとはいえ，ToolBook本体の395ドルという価格は，決して低価格帯のパッケージであるとはいえない点が気になる．

（樋田　亮）



画面B　サードパーティ製HyperCardスクリプトコンパイラ「CompileIt！（Heizer Software，99ドル）」の画面．コンパイラ自体がスタックウェアの形式をとるもので，HyperTalkのスクリプトからXCMD／XFCNを作成する．

SPARCstation SLC

ディスクレスで低価格それでいて高性能

㈱日本サンマイクロシステムズが，ディスクレスではあるものの低価格なワークステーション「SPARCstation SLC（以下，SLC）」を発売した．本体とディスプレイを一体化したことにより，省スペース性も実現している．最近では，Lotus 1-2-3などのアプリケーションもSPARCstation用として発売されており，ワークステーションが事務分野で使われる機会も多くなってきている．SLCは価格，サイズの点でそのような用途に向いているマシンといえる．今回は，そんなSLCをディスクレスマシン独特の使い方も含めて紹介する．

ディスプレイ一体型で省スペース

SLCは，“Super Low Cost”の名前のとおり，RISCチップCPUを採用するワークステーションとしては，これまでになく低価格である．本体，ディスプレイとOSのライセンス料込みで100万円を切る99万5000円という価格を実現した．その仕様をこれまでの主力機種，SPARCstation 1（以下，station 1）と比較する形で**表1**に示す．

CPUとFPUには，それぞれクロック周波数20MHzのSPARC IU，SPARC FPU（WEITEK 3172）を採用している（**写真1**）．処理能力は，12.5MIPS，1.2 MFLOPSであり，浮動小数点演算が1割ほどstation 1より遅くなっている．

メインメモリは，標準で8 Mbytesで，メモリ形状はstation 1などと同じSIMM（Single In-line Memory Module）を採用している．メモリは，本体内に4 Mbytes単位で最大16Mbytesまで増設可能だ．増設方法は，ビスを4本外し，本体カバーを開け，さらにビス2本を外すとメインボードを取ることができ，そのメインボードにSIMMを差し込むだけでよい．メモリボードは，サードパーティ製の安いものも使うことが可能だ．

低価格は，ハードウェアの拡張性を犠牲にすることによって実現した．主な変更点は，システムスロット（SBus）がないこと，パラレルポートがないこと，シリアルポートがRS-232Cで1個のみとなっていることだ（station 1はRS-423）．ディスプレイと本体を同じパッケージに収めたこともコストの低減に役立っている．

外見は，システムボードがディスプレイと同じ本体の中に入っているため，ディスプレイから直接キーボードがのびているように見える（**写真2**）．ディスプレイそのものは，17インチ，解像度が1152×900ドットの標準的なSunの白黒CRTディスプレイである．

キーボードは，Type-4キーボードで，ASCIIタイプとJIS（JLE）タイプのいずれかを選ぶことができる．また，マウスもすでに標準となった光学式の3つボタンマウスで，キーボードに接続する．光学式であるため，専用のマウスパット上で動かす必要がある．

ハードディスクを内蔵することはできない．ハードディスクを使いたい場合には，SCSIインターフェイスに外部機器として接続することになる．最大で4台，

表1 SPARCstation SLCの仕様

	SPARCstation SLC	SPARCstation 1
CPU	SPARC IU（クロック周波数20MHz）	
数値演算コプロセッサ	WTL3172（クロック周波数20MHz）	SPARC FPU（クロック周波数20MHz）
MIPS値	12.5MIPS	12.5MIPS
メインメモリ	8 Mbytes（最大16Mbytes）	8 Mbytes（最大16Mbytes）
フロッピーディスク	―	3.5インチ（1.44Mbytes）
ハードディスク	SCSIで接続可能	104Mbytes
ディスプレイ	1152×900ドット	CRTディスプレイ（オプション）
標準インターフェイス	Ethernet，SCSI，RS-232C×1，オーディオ	Ethernet，SCSI，RS-423×2，オーディオ
拡張スロット	―	SBus×3
標準ソフトウェア	SunOS 4.0以上	
サイズ	409（W）×395（D）×367（H）mm	409（W）×409（D）×71（H）mm
重量	16.8kg	4.5kg
消費電力	200W	85W
価格	99万5000円	186万5000円～

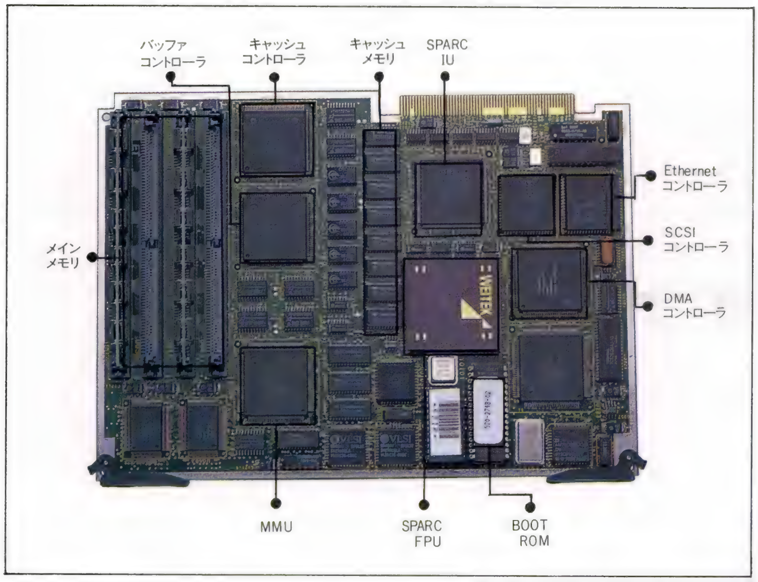


写真1　SLCのメインボード．このボードが本体の後面に縦に入っている．

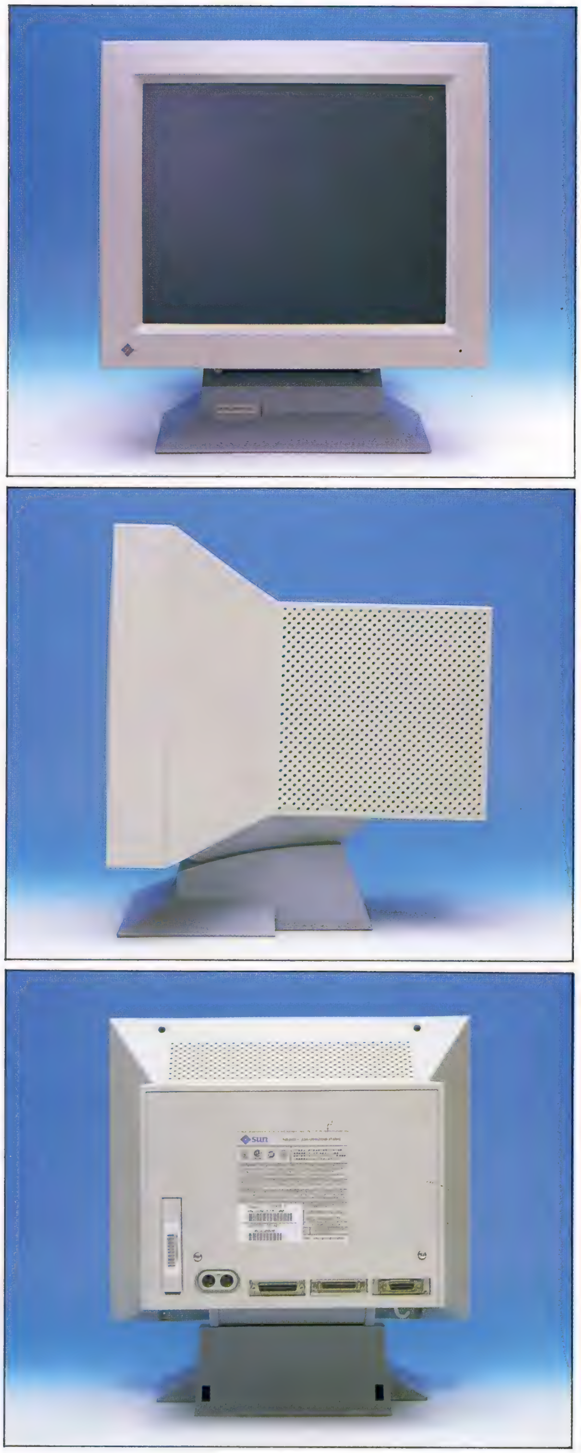


写真2　SLCの正面，側面，後面．ほとんどディスプレイだけといえるデザインだ．そのため，拡張性はほとんどなく，最低限のインターフェイスしか装備していない．

2.6Gbytesまでが接続可能で，カートリッジテープドライブ（CMT）やCD-ROMドライブも接続できる．そのほか，一般的なSCSIデバイスならば接続可能だ．残念ながらフロッピーディスクドライブは，サポートされておらず接続することはできない．

SunOSはRelease4.1にバージョンアップ

OS は SunOS　Release4.0.3とRelease4.1を選択して本体にライセンスされる．4.0.3はJLEを使うことができるが，4.1のほうは今のところはまだ日本語化されていない．

SunOS4.1は，4.0とバイナリ互換で，SystemV Release 4（以下，SVR 4）とのソース互換性を実現している．また，POSIX，X／Openとのコンパチビリティも考慮されている．カーネルなどのチューニングにより，速度もかなり改善された．バイナリ互換ということもあり，4.0.3からの移行は抵抗なく行なえると思われる．4.1の日本語版のJLEバージョンは，9月中旬頃にリリースされる予定である．こちらも手元に届きしだいレポートしたい．

そもそもSVR 4は，これまでのSystemVと4.3BSDを統合したものである．SunOSはもともとBSDにSystemVの機能を付加したものなので，それほど大幅な変更は必要なかったものと考えられる．

SystemV互換だとはいっても，あくまでソース互換である．ディレクトリ構成や，コマンドオプションなどは，従来のSunOSを踏襲している．

インストールの方法は，非常に簡易化されている．一緒にハードディスクを購入すれば，そこにあらかじめOSの主要部分がインストールされてくる．最初の起動時に，sys-configというコマンドが自動的に実行され，ホスト名，ネットワークアドレスなどの基本的な設定はメニュー形式で行なえる．

また，テープやCD-ROMによるインストールも簡単にできるようになった．初めに必要なミニルートを，ワンステップで作成できるのだ．ディスクレスクライアントとして，またサーバーとしての設定は，add_client，rm_clientというコマ

ンドを使用する．

キーボードの配列も容易に変更できるようになった．キートップに関係なく，loadkeysコマンドを使用して，キーテーブルを書き換えられる．キーテーブルのテンプレートは，多くの種類が用意されているが，現状でカナキーボード用のものは見当たらなかった．

そのほか国際化についても，改善がなされている．ほとんどのコマンドで8ビット文字を使うことができる．また，カーネルやライブラリでも問題なく8ビット文字が使用可能だ．日本語など，ラテン文字セット以外のものについても，ローカルテーブルを設定することにより使える．ただ，まだ日本語用のテーブルは用意されていない．新機能としては，オンラインヘルプ，プログラマブルアラームなどがある．また，Type-4キーボードのサポートもOS同様に行なわれている．

拡張性を抑えて低価格化

さて，価格を下げるために犠牲となった拡張性はどうだろうか．外部に出力されるインターフェイスは，オーディオ，キーボードを除くと，Ethernet，RS-232CおよびSCSIがそれぞれ1ポートずつである．

SCSIポートが1個であることに問題はない．SCSIデバイスは，基本的にデイジーチェーンでつなげるので，最大7デバイスまでこのポートにつなぐことができる．ただ，ハードディスク，CD-ROM，CMTなどのそれぞれのID番号が決まっているので，この点は注意が必要だ．

Ethernetが1ポートであることも，通常は問題にはならないだろう．2nd Etherをこのタイプのワークステーションに接続し，ルーターなどに使用するということは考えられないからだ．

シリアルポートの数は，場合によっては問題になる．スタンドアロンのマシンとして使う場合，ここにモデム，プリンタ，ターミナルなどをつなげて使うことが考えられる．どれをつなげるか，頭の痛いことになるかもしれない．

最大の問題は，SBusスロットが1個もないことである．これまでのSunシリーズのマシンには，このスロットが必ず複数あった．フロッピーディスクドライブやシリアルポートの増設，Ethernetポートの増設などは，このスロットを利用して行なっていたわけだ．SLCは，必然的にパラレルポートやフロッピーディスクドライブを付けることができないことになる．

SPARCシリーズのソフトウェアには，フロッピーディスクによって供給されているものもある．また，データをフロッピーに収めて持ち歩くことができないことにもなる．プリンタもシリアルインターフェイスを持つものしか接続できない．

製品のコンセプトとしては，ディスクレスということもあり，ネットワークに接続されることを前提としているが，このような低価格マシンは，さまざまな状況で使われることが考えられる．たとえば個人が，パソコン代わりの完全なスタンドアロンマシンとして使うこともあるだろう．そのときには，Ethernetなどいらないが，フロッピーディスクやシリアルポートはもっとほしいかもしれない．そこで，用途の限定されたインターフェイスは設けず，汎用のシステムバススロットのみをいくつか付けておくことを提案したい．オプションにより，必要なインターフェイスを選べるようにすれば，自由度が広がるのではないだろうか．

sys-configで初期設定

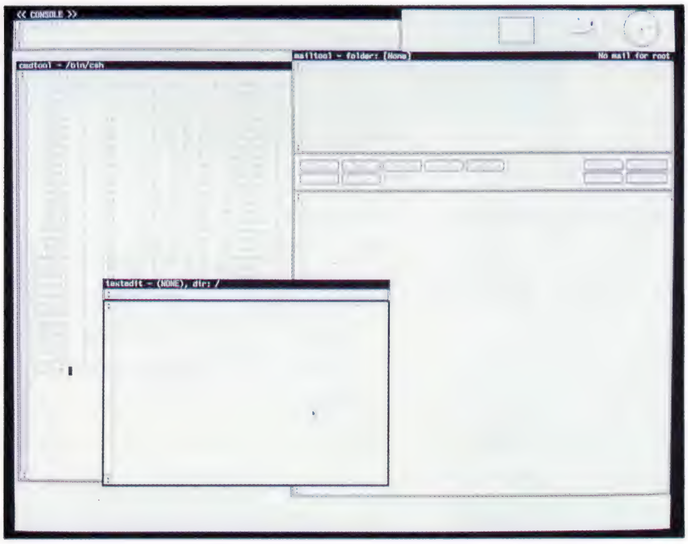
今回試用したマシンは，SLCの標準構成に，オプションの104Mbytesのハードディスクとcmtを付けたものだ（ただし，今回はCMTは使用しなかった）．メモリも標準の8Mbytesのままである．

起動前のハードウェアの接続は，ほとんど手間取ることなくできた．若干のディップスイッチなどの設定とケーブルの接続だけだ．

104Mbytesのディスクは，あらかじめパーティションが切られており，スワップ領域も確保されていた．そして先ほども述べたようにSunOSはインストールずみである．スタンドアロンならば，スイッチを入れるだけで使い始めることができる．

電源を入れると，メモリなどのテストのあと，いきなりネットワークからブートしようとした．つまり，このマシンはクライアントのディスクレスワークステーションとして使われることを想定しているようだ．つまり，ディスクレスクライアントとして使う場合には，Sunのサーバーにつなげるだけで，ほかの設定は必要がないということだ．ちなみに，ディスクレスクライアントとは，ルートファイルシステム，スワップをすべてサーバーに持ち，ブートもサーバーのディスクから行なう形態である．

今回は，ハードディスクを装着しているので，ROMモニタのスイッチを変更し，ハードディスクから起動できるよう



画面1　SunViewの画面．高解像度のCRTということもあり，大変見やすい．ただ，映り込みが激しいのが欠点である．

にする必要がある．欲をいえば，最初の起動時だけ，どこからブートするか聞いてほしいものだ．

SLCのROMモニタはstation 1と同じくフルセットのFORTHで書かれている．ROMモニタから，ひと通りのことはできる．驚くべきことに，コマンドのヒストリ機能まである．いったい誰が使うのかは不明だが．

表2にROMモニタで設定できるレジスタの一覧を示した．ご覧のように，ブートの方法，I／Oポートなどを，ROMモニタからsetenvコマンドで設定できる．

さて，ハードディスクからのブートを行なう．すると，ログインプロンプトが出る前に，自動的にsys-configが実行される．フルスクリーンのメニューにより，ホスト名，標準時，現在の時刻，ネットワークにつながっている場合はアドレスとNIS（**注1**）の設定をする．設定が終わると，自動的に通常のUNIXのブートが行なわれ，ログインプロンプトが出る．このsys-configは，当然のことであるが，初めて起動したときのみ実行される．

UNIXワークステーションとしては，このsys-configは従来にはない斬新なものだ．これまでは，いったんシングルユーザーモードで起動して設定し，起動し直すなどの作業が必要だった．それに比べると，手軽にインストールすることができる．ただ，残念ながらまだ完全ではなく，ネットワークのルーティングやブロードキャストの設定などは，手でやるしかなかった．

少し大きめのネットワークだと，1本のネットにすべてのマシンがぶらさがっているというわけにはいかない．サブネットに分けて，ルート設定を行なうのが一般的だが，今のところsys-configでは対応できないようだ．

だが，全世界がネットワーク接続されようとしている今，遅かれ早かれ改善されることだろう．このsys-configが拡張されて，起動に必要な情報をすべてインタラクティブに設定できるようになれば，ますます複雑化していく環境の中で，強力なツールとなりうる．今後に期待したい．

表2　ROMモニタで設定できるレジスタ

Parameter Name	Value	Default Value
selftest-#megs	1	1
oem-logo	5a 5a 5a 5a 5a 5a 5a 5a 5a ...	
oem-logo?	false	false
oem-banner		
oem-banner?	false	false
ttyb-mode	9600,8,n,1,-	9600,8,n,1,-
ttya-mode	9600,8,n,1,-	9600,8,n,1,-
sbus-probe-list	0123	0123
fcode-debug?	false	false
screen-#columns	80	80
screen-#rows	34	34
boot-from-diag	sd(0)vmunix	le()vmunix
boot-from	vmunix	vmunix
auto-boot?	true	true
watchdog-reboot	false	false
input-device	keyboard	keyboard
output-device	video	video
keyboard-click?	false	false
keymap		
sunmon-compat?	true	true
security-mode	none	none
security-password		
security-#badlogins	0	0
sd-targets	31204567	31204567
st-targets	45670123	45670123
scsi-initiator-id	7	7
ttyb-rts-dtr-off	false	false
ttyb-ignore-cd	true	true
ttya-rts-dtr-off	false	false
ttya-ignore-cd	true	true
hardware-revision	xxxx	
last-hardware-update		
testarea	0	0
mfg-switch?	false	false
diag-switch?	true	true

操作感はstation 1と変わらない

実際の使用感は，ごく普通のSPARCシリーズのワークステーションと変わらない．速度も十分に速く，快適である．スピードの目安は，ベンチマークテストの結果を見ていただきたい．浮動小数点演算がスペック通り若干遅いことを除けば，station 1と同等のパフォーマンスだ．

これもstation 1と同じなのだが，あいかわらずコンソールの画面出力は非常に遅い．これはSunの伝統といってもよいくらいだ．ウィンドウシステムよりもコンソールのほうが遅いというのは何とも不思議な感じがする．

Sunのマシンはどれも使い心地は同じである．Sun 3／60とこのSLCとは，何世代も違うマシンだが，スピードを除けば，ほとんど区別することなく使用できる．これは，進歩がないというよりは，完成された安心感といったほうが正しいだろう．

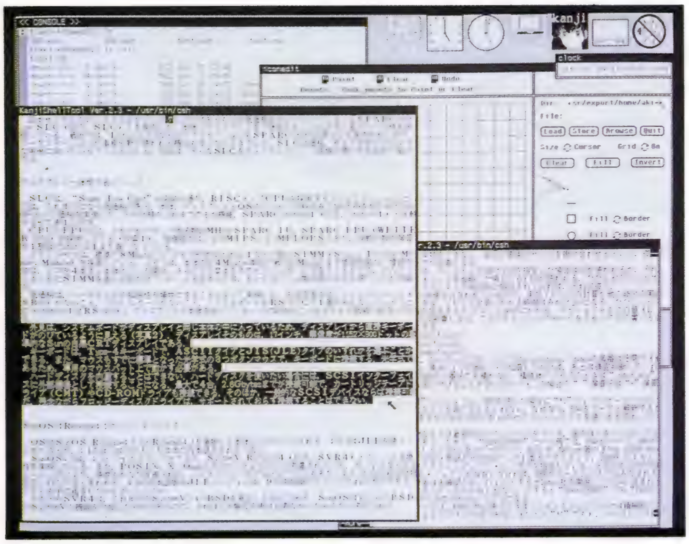
SLCは，従来のディスプレイのサイズにディスプレイと本体を押し込めている．画面もまったく問題なく，いかにもSunのディスプレイである（**画面1**）．ただ，あいかわらず室内照明の映り込みが気になる．日常的に長期間使うのであれば，フィルタを付けるなどの工夫が必要だろう．できれば，ノングレア処理などを施してほしいところだ．

放熱は，本体に無数の穴を設けることによって行なっている．ファンがなく，とても静かである．ただ，発熱量は大きく，本体上部は長時間使用していると熱くなり，やや不安を感じる．

SunOS4.1で日本語を使う

SunOS4.1には，日本語サポート機能がない．そのため，標準では漢字の表示な

注1　NISはネットワークのホストやパスワードなどのデータベースの管理のために使われるシステム



画面 2　KanjiShellToolsを使って漢字を表示した．MS-DOSの文書を表示するには，事前にシフトJISコードからEUCコードに変換しておく必要がある．

どはできない．だが，SunView上で漢字を表示することは可能だ（**画面 2**）．

JUNETやユーザーズグループから，ライセンスフリーの漢字フォントk14や，SunViewのKanjiShellToolという漢字表示可能なターミナルエミュレータなどを入手することができる．どちらも，マニュアルに従って，すぐにインストールできた．

また，Wnnなどの日本語入力FPを入手すれば，JLEがなくてもひと通りの日本語の入力と表示ができる．

この記事も，KanjiShellToolと，リモートマシンのNEmacsを使用して書いた．日本語入力FPは，Emacs-Lispによって書かれたSKKを使用した．

なお，k14はH．Tachibana氏，KanjiShellToolは矢野達朗氏，SKKは佐藤雅彦氏の作である．

X Windowをインストールすることによっても，日本語を使用することができる．X Window上で日本語を使えるツールもいろいろあるため，かなり自由に使えるようになるはずだ．

X Windowのインストールは，ディスクスペースの関係で見送ったが，機会があればレポートしたい．

ただ，このようにしてSunOS4.1上で日本語を使えるようにしたとしても，日本語版SunWriteなどのアプリケーションは使用できないので注意が必要だ．

ディスクレスをどう使うか？

SLCに限らず，Sunワークステーションは，ネットワーク機能が充実している．ネットワークを通じてのブートや，NFS，RFSのサポートにより，さまざまな利用形態が考えられる．

まず，SLC本来のディスクレスクライアントとしての使用方法について考えてみる．

この方法は，大きなディスクを集中的に管理すれば全体を管理できるため，メンテナンスが容易だという特徴がある．ただし，スワップが起きるたびにネットワークに大量のデータが流れるので，ネットワークの負担は大きくなる．なるべくスワップを減らすためには，クライアント側のメモリは拡張しておくことが望ましい．

ほとんどのファイルをローカルに持つ基本的な方法もある．もっともネットワークやほかのマシンに負担をかけることはないが，同じマシンが複数台あるときには，同じファイルを持つ必要があるなど無駄が多い．また，集中管理には向かず，リモートログインした際に環境が変わって使いにくい．

これらの間を取って，ルートファイルシステムとスワップだけをローカルに持ち，バイナリやライブラリ，ソースなどをサーバー上で共有する方法もある．この方法が最も効率がよく，ディスクをあまり使わずにすむ．また，ネットワークへの負担もさほど重くなく，データの共有が容易である．

ほかのホストへのリモートログインが主であれば，ディスクレスで十分である．X端末と似たような利用方法だ．

ただ，X端末と違ってホストの機能を肩代わりできるので，その分ホストの負担は軽減する．だが，中途半端にクライアント上で作業を行なうと，ネットワークの負担は増大してしまい，メリットがなくなってしまう．

また，数人共同で行なう作業の場合は，ルートファイルシステムのみローカルに持つ方法がベストである．ソースやデータの共有が簡単だし，パフォーマンスもスタンドアロンで使うのに比べてそれほどは落ちない．

このように，どのような管理形態をとるのがベストかは，利用形態によって異なる．事前に十分検討することが必要だ．いずれの状況でも，SLCは柔軟に対応することができる．ただし，X端末はSLCに比べても低価格であり，サーバーマシンに多くの端末をつなげるような場合には，コストパフォーマンスが高くなることも考えられる．

誰にでも使えるワークステーション

SLCの目玉はその価格にある．それでいて，これまでのSunワークステーションと完全な互換性を持っている．その速度，機能はワークステーションの平均レベルをはるかにしのぎ一流である．

station 1とほぼ同じ性能でこの価格を実現できたのには，SLCと同時に発売されたstation 1＋の発売も影響している．実質的には，SPARCstationシリーズの大幅な値下げだといえよう．

OEM製品も含めてSunのワークステーションは，独自の方向性を持っている．それは，誰にでも使えるワークステーションということだ．たとえば，sys-configを使えばほとんど知識がなくても，とりあえずインストールして使い始めることはできる．

SunViewという，シンプルでグラフィ

カルなインターフェイスも標準装備しており，面倒な設定を行なわなくても，その日からウィンドウ環境で仕事ができる点も魅力だ．

また，SunにはSunWrite，SunDraw，SunPaintやLotus 1-2-3などのアプリケーションも整いつつある．パソコン用の有名なワードプロセッサなどもリリースされる予定である．SunViewも，マルチウィンドウのアプリケーションランチャーとして便利になってきた．ただ，これらのインストールにはオプションのCMTを購入する必要がある．

こうなると，SLCはこれまでのワークステーションの役割——大規模な開発用マシン——にとどまらず，さまざまな用途が考えられる．会社や家庭のパソコンを置き換えてしまうことも不可能ではないのだ．

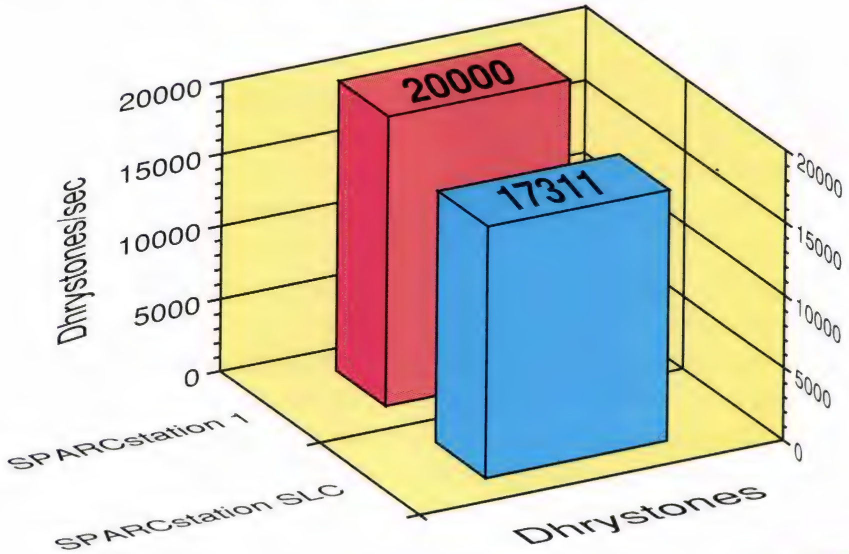
一方では，本格的な使用にもさまざまなソフトウェアで対応している．OpenWindows やXViewなどのネットワークウィンドウシステムやSunNetなどにより，本格的にネッワークを利用した開発環境も構築できる．

惜しいところは，フロッピーディスクが装着できないこと，ハードディスクが内蔵できないこと，拡張性に乏しいことだ．できれば，SBusの搭載などstation 1に近いスペックのディスプレイ一体型マシンもほしい．

個人用マシンからネットワークの中枢となるマシンまでを同じアーキテクチャで統一しようというSunの構想は，まずまず成功しつつあるようだ．今後，多様化しつつ拡大していくワークステーション市場が，どのような展開を見せるのか注目させられる．

(上野和明)

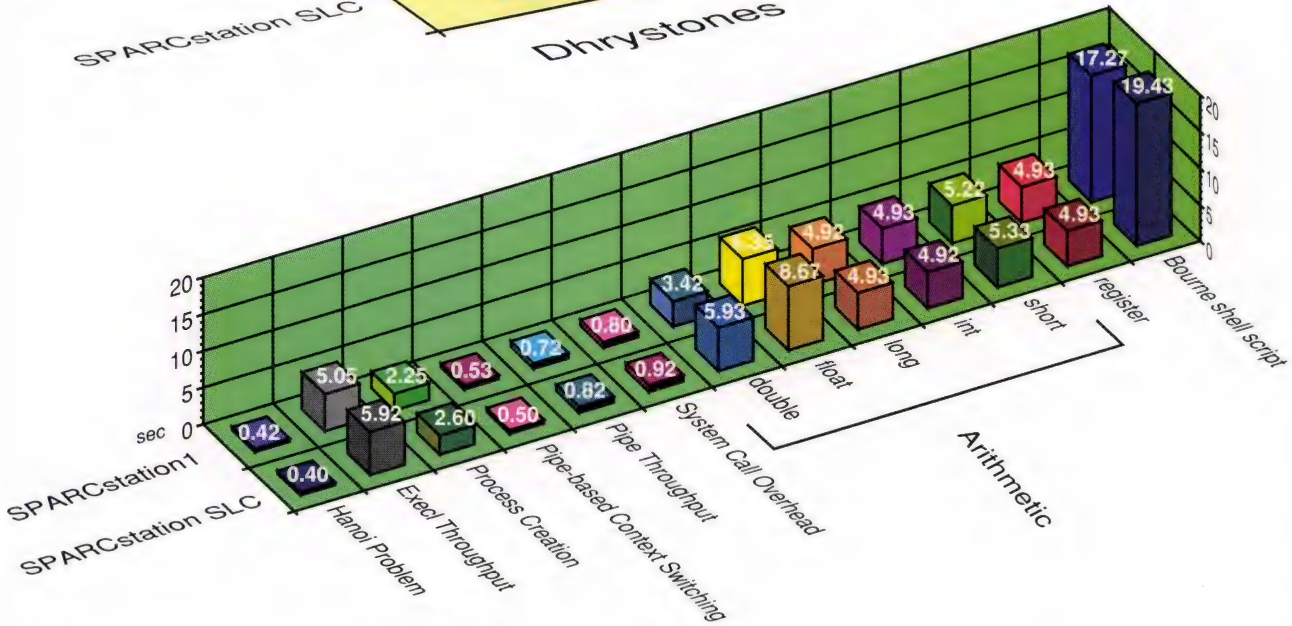
ベンチマークテストの結果



下の図は，SLCのベンチマークテストの結果である．今回も，先月同様BYTE's UNIX Benchmarksを使用した．先月同様，station 1の結果を併記する．

結果としては，station 1とほぼ同じことになった．アーキテクチャ，条件が同じなので，当然のことである．浮動小数点演算は，FPUのスペック通り，station 1よりも劣る．

station 1のOSは4.0.3，SLCのOSは4.1である．OSのバージョンアップによるパフォーマンスの向上は，このテストにおいては見られなかった．条件を変えれば，結果も変わるかもしれない．



Model-10J

QMS ColorScript100

日本語PostScript搭載のカラープリンタ

㈱ドットウェル ビー・エム・エスは，日本語PostScript対応のカラー熱転写プリンタを発売した．『QMS ColorScript100 Model-10J（以下10J）』と『同40J』である．この２つの製品は，米QMS社の従来のColorScript100シリーズを日本語PostScriptに対応させたものだ．

解像度300dpiのPostScriptプリンタ

10Jと40Jの違いは，10Jで使用できる用紙サイズがＡ４およびＡ４変形であるのに対し，40JはＡ４とＡ３の用紙が利用できる点だ．そのため，本体のデザインはまったく異なったものになっている．ただし，どちらも解像度300dpiの熱転写方式であり，プリンタとしての機能に基本的な相違はない．

PostScriptインタープリタ「RIP(Raster Image Processor)」を含むコントローラ部には，クロック周波数16MHz(10J)／16.67MHz（40J）のMC68020を搭載し，RAMは10Jが8Mbytes，40Jが12Mbytesを標準装備している（**表１**）．今回は，10Jを中心に紹介を行なうが，ハードウェアに関する直接的な事柄以外は両機種に共通のものと理解してもらいたい．

セットアップはいたって簡単

10Jと40Jはどちらも，プリンタ本体とPostScriptの日本語フォントを収めた40MbytesのSCSIハードディスクで構成される．セットアップはいたって簡単で，電源ケーブルを接続するほか，ハードディスクとプリンタ，CPU本体とプリンタのそれぞれをケーブルで接続するだけだ．一般にSCSIハードディスクは，50ピンのアンフェノール型コネクタで接続するものが多いのだが，付属のハードディスクは25ピンのD-subコネクタで接続するものだった．インターフェイスは，利用するコンピュータに応じて，RS-232C，セントロニクス(パラレル)，AppleTalkのいずれかを選んで接続することができる．そのため，Macintosh(以下，Mac)，PC-9801，IBM PCなどのほか，ワークステーションにも接続可能だ．今回はその中で，PostScript対応のグラフィックソフトが充実しているMacを利用してテストを行なったため，AppleTalkで接続した．インターフェイスとメディア（コート紙を含む普通紙とOHPフィルム）の選択は，フロントパネルにあるタッチ式のキーで行なう．

10Jの電源を投入すると，60秒程度のウォームアップの後，自己診断テストを兼ねたレポートが印刷される．レポートには，PostScriptインタープリタのバージョンや標準搭載のPostScriptフォント，通算の印刷枚数，アクティブなインターフェイスなどの諸情報に加えて，C(Cyan)，M(Magenta)，Y(Yellow)，K(Black)各色が色見本として打ち出される．

電源を投入して最初に気づくのは，その騒音だ．カタログでは55db以下とあり，レーザープリンタと比較しても数値の上ではさほど違いがないのだが，アイドリング中も常に高音の騒音が続き，精神的にかなりこたえる．この騒音は，10Jの本体下部に取り付けられた冷却ファンのためのようだ．改善してほしい点である．

消耗品のインクシートロールには，C，M，Y，Kの４色のものと黒を除く３色のもの，黒色のみのものの３種類が用意されている．４色が85ページ，３色が100ページ，黒色が300ページを１本のロールで印刷することができる．価格は３種類とも１本5500円(税別)．ロールはプラスチック製のカートリッジに収めて本体に装

表１　ColorScript100 Model10J／40Jの主な仕様

	Model 10J	Model 40J
プリント方式	熱溶融転写記録方式	
解像度	300dpi	
用紙	A4／A4変形	A3／A4
プリント速度	70秒／4／4色 85秒／A4変形／4色	60秒／A4／3色 105秒／A3／3色
CPU(クロック周波数)	68020(16MHz)	68020(16.67MHz)
RAM	8Mbytes	12Mbytes
ROM	1Mbytes	
インターフェイス	Apple Talk,RS-232C,セントロニクスパラレル	
サイズ　W×D×H	430×631×288mm	531×564×599mm
重量	29.5kg	37.1kg
騒音	55dB以下	
消費電力	400W	650W
価格	295万円	495万円

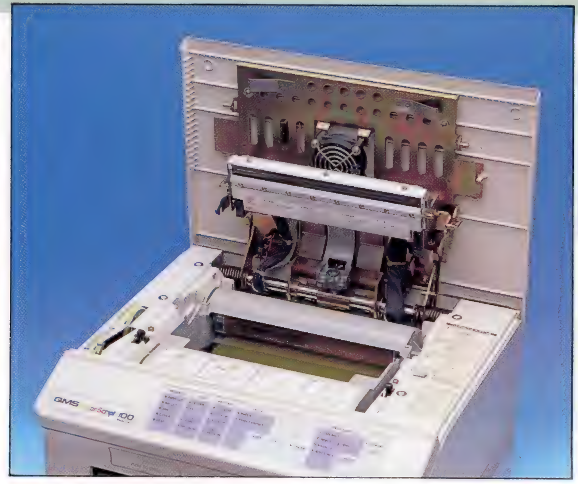


写真1　本体内部に見える緑色のプラスチックの容器が，インクシートロールを装着するカートリッジだ．カートリッジはそのまま持ち上げて，簡単に外すことができる．

写真2　4色のインクシートロール．先頭から順にC，M，Y，Kのインクブロックが並んでいる．1ページの印刷には，必ず4色分のインクを消費する．

着するため，交換作業は簡単にすむ（**写真1**）．メンテナンス性は非常に良好だ．ちなみに，4色で85ページというのは，10Jのペーパーカートリッジに収めることのできる用紙枚数と同じだ．

4色のインクシートロールは，先頭から順にC，M，Y，Kのインクのブロックで1セットになっている（3色も黒を除いて同様：**写真2**）．そして1ページの印刷には，必ず1セットのインクを消費する．つまり，たとえ黒色のみの印刷をしても4色分のシートが送られて，他の3色は無駄になってしまうわけだ．そのため通常使用すると思われる4色インクシートロールでは，1ページ当たり紙代を除いて約65円のランニングコストとなる．決して安いとはいえない．このあたりの事情は，熱転写方式のカラープリンタに共通の短所といえるだろう．

2書体を搭載した日本語PostScript

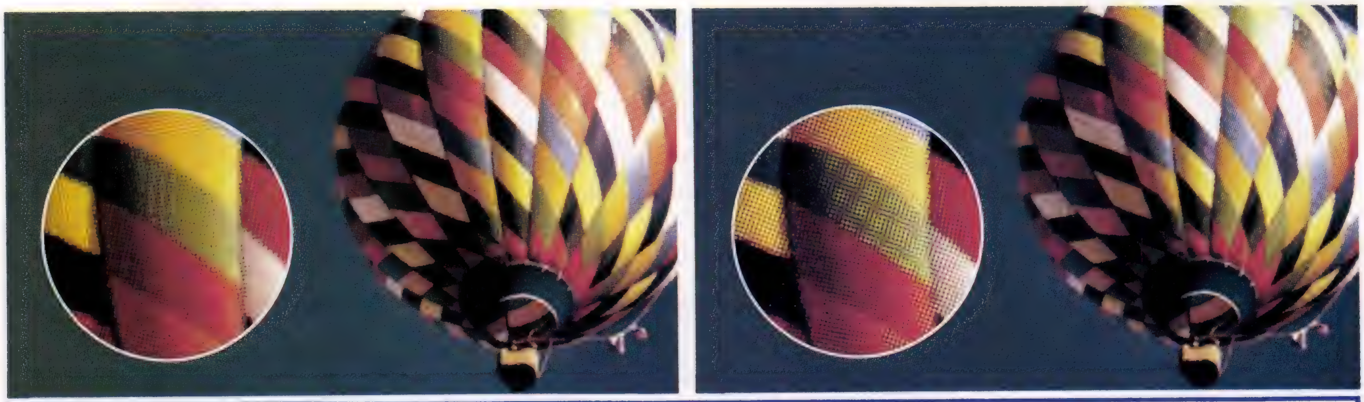
日本語PostScriptは，昨年Apple社のLaserWriter NTX Jや日本電気のPR-602PSに搭載されたのをはじめとし，現在では国内数社から対応プリンタが発売されている．それらの大半がレーザープリンタであり，カラー出力が可能な日本語PostScriptプリンタは皆無であった．そのため，プレゼンテーション資料用などでカラーグラフィックスと日本語を併用した出力を必要とする場合には，グラフィックス部分だけを英語版のColorScript100シリーズで出力し，LaserWriter NTX-Jで日本語文字部分のみを重ねて出力するなど苦心しているユーザーもいたようだ．

10J／40Jでは日本語PostScriptに対応しているために，そうした手間をかけることなくカラーグラフィックスと日本語の混在したページ出力を得ることが可能になっている．付属の40Mbytes容量の外部ハードディスクには，日本語PostScriptでは標準のモリサワ製リュウミン-KL（細明朝体）と中ゴシックBBBの2つの日本語フォントが搭載されている（**印字例1**）．また，ローマン系フォントは，Avant Garde，Helvetica，Timesなど35書体をROMで搭載している．Adobe社やサード

印字例2　日本語版Illustratorによるアートワークの出力例．花鳥風月の文字も，日本語フォントのリュウミンに変形，色指定を加えたものである．

印字例1　リュウミン-KLと中ゴシックBBBの印字例．熱転写方式であるため，レーザープリンタに比較すると，文字の縁に細かなバンプ（凹凸）が目立つ．

リュウミン-KL	中ゴシックBBB
9p 月刊スーパーアスキー	9p 月刊スーパーアスキー
24p 月刊スー	24p 月刊スー
月刊	月刊



印字例 3　各色でスクリーン角度が同一だと，イメージは全体に黒っぽくくすむ．また不適切なスクリーン角度を設定すると，網干渉が生じイメージの再現に支障をきたす．

パーティから市販されているPostScriptフォントを，ハードディスクにダウンロードして使用することももちろん可能だ．

PostScriptではローマン系の書体が数百種類用意されていることを考えると，日本語フォントの標準で2書体というのはいかにも少ない．すでにモリサワから3書体，サードパーティからも数書体発売されているものの，さらなる充実が望まれる．

カラーPostScriptによる出力の実際

それでは，実際の出力例を見ながらカラー出力の方法について解説しよう．

印字例 2 はAdobeの日本語版Illustratorで作成したアートワークの出力例だ．デザインされた漢字文字は，標準搭載のリュウミンに長体や斜体など変形を加え，色指定を行なった．文字の回転，変形などに柔軟に対応するPostScriptの能力を，あらためて確認できるだろう．

PostScriptではカラー(グレースケール)の中間調を，一般の印刷と同様に網点のハーフトーンスクリーンで表現している．**印字例 2** でも中間色が4色の小さな点の集まりで表現されていることが分かる．

ハーフトーンスクリーンの設定は，ソフトウェアで変更することが可能だ．ただし，10Jでは4色のインクシートロール使用時の標準値で，各色60lpi (lines per inch：線密度)，スクリーン角度はC15度，M75度，Y30度，K45度となっている．カタログではこの設定値で17,576色を表現できるとなっている．特にソフト側で線数などを指定しないかぎり，この値によるスクリーンで出力される．参考までに **印字例 2** の出力には，Macで印刷の実行ボタンをクリックしてから，ペーパーが完全に排出されるまで5分11秒かかった．LaserWriter NTX-Jで同じデータを出力するのに3分38秒であることを考えると，10Jの速度は十分に及第点を与えることができる．

階調表現力と実質解像度

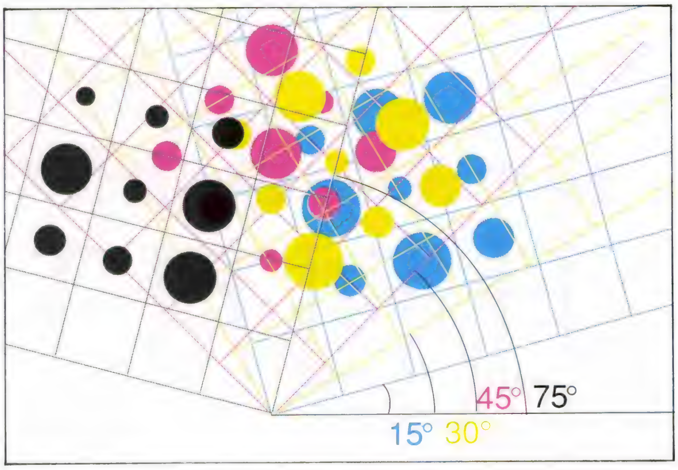
ハーフトーンスクリーンについて，もう少し解説を加えておこう．

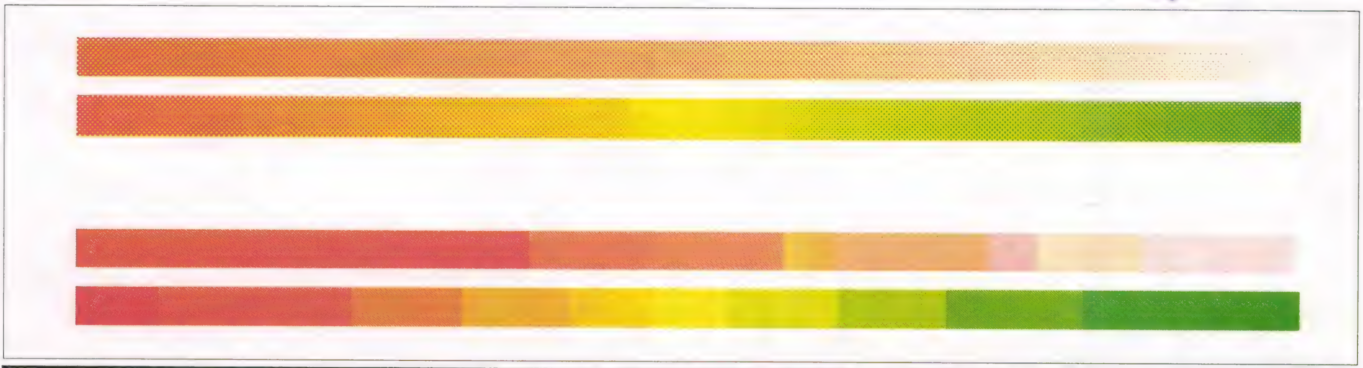
ハーフトーンスクリーンとは，色の濃淡（階調）を，網点の大小に置き換えて表現する方法のことである．線密度とは1インチ当たりにいくつの網点の列を配置するかという単位で，スクリーン角度とは網点の列がなす角度のことである．

一般の印刷にもいえることだが，PostScript出力においてスクリーンの線密度の設定と表現できる中間調，実質解像度には密接な関係がある．300dpiモノクロのプリンタを例にとって考えてみる．2階調ならば300dpiの解像度がそのまま活かせるのだが，10階調を表現するためには縦横に3×3＝9個のドットを1つの網点とする必要がある．すると実質解像度は300÷3＝100dpiとなる．より多くの階調を表現するためには，1つの網点を構成するドットの数を増やす必要があり，実質解像度が低下してしまう．

さらに10Jのようなカラープリンタでは，C，M，Y，Kの4色のインクで1つの中間色を表現するため，各色の網点が不必要に重ならないように，位置を相互にずらす必要がある．そのため，より実質解像度が低下してしまう．各色の網点の位置をずらすのは，4色の網点が1カ所に集中すると，仮に透過性のあるインクであっても，再現されるイメージは全体に黒っぽくくすんだ感じになってしまうためだ．この網点の位置を決定する要素がスクリーン角度で，0～90度の範囲で設定する．スクリーン角度は，適切な値を設定しないと網干渉（モアレ）の原因となる(**印字例 3**)．理論的には，各色30度ずつ角度を変えると，網干渉は最小限になるとされているが，0度と90度は

図1　C，M，Y，(K)による減法混色では各色の網点をずらして配置しないと，イメージ全体に黒みがかってしまう





印字例 4 上側が60lpi, 下側が90lpiに設定した出力結果. 線密度が高くなれば網点は見えなくなるが, 中間調の再現力は大幅に低下することが分かる.

同じになるため, 最も影響の少ないYを15度ずらす方法がとられる(**図 1**). 先に述べたが, 10Jの既定のスクリーン角度が, C15度, Y30度, K45度, M75度となっているのはそうした理由による.

つまり, 10JなどのカラーPostScriptプリンタでは, 色(中間調)表現力と実質解像度は相反する要素なのである. その能力を十分に活用するためには, カラー印刷に関する知識が必要なわけだ.

では, 色表現力と実質解像度が相反することを, 具体的な例で見てみよう.

まず, **印字例 4** はカラーグラデーションの例だが, デフォルトの60lpiで出力したものと, 90lpiで出力したものでは出力結果に大きな違いが認められる. 60lpiでは網点が気になるものの, グラデーションはきれいに表現されている. 対して, 90lpiでは網点は目立たなくなっているが, 表現できている中間色が極端に少なくなっていることが分かる. 60lpiよりもさらに線密度を減らせば, グラデーションに現われる縞は目立たなくなるが, 網点はより目立つようになる.

さらに**印字例 5** は, 24bitカラーでスキャンした写真のビットマップグラフィックスを出力したものだ. 線密度が低ければ, 色は全体にソフトで自然な感じに仕上がるが, イメージのディテールはボケてしまう. 逆に線密度を高く設定すると, ディテールはクッキリとシャープに仕上がるが, 色の変化に少し不自然さが加わる. 写真のように色数が多く, イメージ, 色の変化がともに複雑な画像を出力する場合には, 各色ごとに細かく線密度とスクリーン角度を設定しなくては期待した
リーン角度を設定しなくては期待したクオリティを持つ出力結果を得られない. 見方を変えれば, 線密度／スクリーン角度の設定で異なる雰囲気を持つ出力が得られるのだ. 色の表現力を優先するか, ディテールの克明さを優先するかは, 目的に応じて設定を変更すればよい.

☆

解像度が300dpi程度のプリンタでは, ハーフトーンスクリーンのカラー表現力は, ほかの多くの非PostScript系プリンタで行なわれているディザパターンによるカラー表現力に劣ってしまう. ディザパターンによる方法では, 1つのパターンを構成するドットグループを, 搭載しているインクの各色を取り混ぜて構成できるのに対し, ハーフトーンスクリーンでは各色ごとに単色の網点(ドットグループ)で構成するためだ.

しかし, このことは10Jの位置付けを考慮すれば, 一概に解像度に対するカラー表現力が低いと非難できない. なぜなら, 10Jは一般の印刷におけるカラー画像の4色分解→製版→最終的な出力をシミュレーションするものととらえることができるからだ.

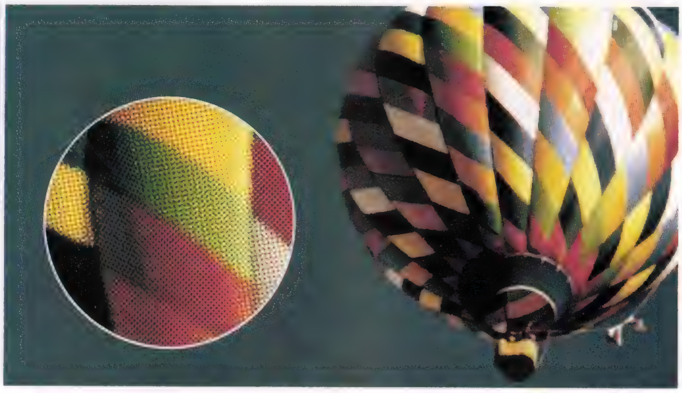
日本語PostScriptに対応したColorScript100シリーズは, デザイン, 印刷, 出版などの分野で活躍しているユーザーにとって, カンプやプレゼンテーション資料の作成に威力を発揮するに違いない.

価格は, ほかの日本語PostScriptプリンタ(モノクロ／300dpi)が90万～100万円前後であるのと比較し, Model-10Jで295万円, 同40Jで495万円というのはかなり高額の部類に入る. しかしそれもカラー出力が可能で, パーソナルユースよりもデザイン分野などのプロユースを想定した製品であることを考慮すると, 納得できる価格ではないだろうか.

(樋田　亮)

製品名 ―― QMS Color Script100 Model-10J/40J
価格 ―― 295万円(10J)/495万円(40J)
問い合わせ先 ―― ドッドウェル ビー・エム・エス 東京OS営業部 Tel:03-555-9831

印字例 5 写真など色数が多くイメージも複雑なグラフィックスの出力には, 各色ごとに線密度, スクリーン角度の設定を変えることで仕上がりが異なってくる.



Hardware=Software+Hardware
NonStopコンピュータのための GUARDIAN90

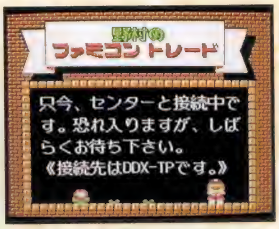
特殊な用途向けに特化したコンピュータが，使われだしている．当然これらのコンピュータには，特殊なハードウェアをサポートする特殊なソフトウェアが必要になる．これをプログラマの視点で紹介していく．

佐倉 はじめ

現在存在しているコンピュータシステムの半数近くはOLTP（オンライントランザクション処理，リアルタイムでのアトミックなデータベース更新）が用途であるといわれている．バンキング，POSといった安定した運用が絶対条件であるOLTP業務においては，高いトランザクション処理能力だけでなく，システム全体の連続稼働を保証するフォールトトレランスが要求される．

タンデムのNonStopシステムはそういった市場のニーズをくみとって，OLTPに目的を絞って開発されたアプリケーションスペシフィックなメインフレームコンピュータである．完璧なフォールトトレランス，またデータベースの完全性や，プロセッサの数が処理能力と正比例する直線的拡張性が他のメインフレームと一線を画す．処理能力は最上位機種のCyclone（**写真上**）で，16プロセッサ構成時400TPS（トランザクション／秒）となっている．

NonStopシステムのこういった特徴はマルチプロセッサアーキテクチャによるところが大きい．2プロセッサが最小構成で，最大16のプロセッサが互いに他のプロセッサのバックアップを行ないながら自分のプロセスを実行することで，フォールトトレランスを実現している．こうしたアーキテクチャが効果を発揮するにはOSの役割が重要だ．そこで専用OS“GUARDIAN90”の働きを中心に，NonStopシステムの特徴的な部分を解説していく．



◀ファミコントレードのホストはNonStopVLXシステムである．PATHWAY環境でCOBOLを使ってプログラム開発されたサービスと，高速レスポンスのためTAL言語で開発されたサービスとの，大きく分けて2系統を持つOLTPシステムとなっている．

◀運転中のCycloneから，いきなりCPUボードを引き抜く．システムは何の問題もなく稼働を続ける．このあとボードをもとに戻すと，そのボードはすぐさま動作を開始した．電源のコンセントを引き抜いても，また差せばすぐに運転を再開するという．非常に効果的なフォールトトレランスのデモといえる．

▶雪印乳業の販売流通管理システムのホストにもタンデムNonStopシステムが採用されている．流通，金融といったトランザクションの量が多く，システムダウンが経済的にも，また信用面でも大きな影響を及ぼす用途は今後も増えていく傾向にあり，また既存のシステムの負荷も利用の拡大により増大しつつある．

NonStop ハードウェア

●プロセッサ

タンデム NonStop システムは 2 重系を基本とする疎結合マルチプロセッサ構造を採用している. 最小構成で 2 プロセッサ, 最大16プロセッサまで拡張可能で, 各プロセッサはメモリ, バスコントローラ, I／O チャネル, 電源, 診断機能を個別に備え, それぞれ独立したクロックで作動し, 共有メモリはいっさい持たない. プロセッサ間の通信は Dynabus と呼ばれる 2 重化バスを介して行なわれる. Dynabus の総帯域幅は各機種とも40 Mbps（20Mbps× 2）となっている.

NonStop システムにおけるプロセッサの多重化は, 通常のホットスタンバイによるバックアップと異なり, 常時すべてのプロセッサにプロセスが割り当てられることになるので, パフォーマンスとフォールトトレランスが両立する.

●ディスク

NonStop システムのすべての I／O コントローラはデュアルポート化されており, それぞれのコントローラは 2 台のプロセッサに接続される.

ディスクコントローラには物理的に独立した 2 台以上のディスク装置を接続し, ミラードディスクとして使用する. これにより, 一方のディスクに発生した障害を回避できるだけでなく, アクセスアームが必要なデータの近くにあるディスク装置をアクセスすることで, シーク時間を節約し, システムのパフォーマンスを向上させることができる. なお, 物理的には 2 台のディスクもアプリケーションから見たときにはプライマリ／ミラーの両ディスクで 1 つの論理デバイスとみなされ, 上記のような機能はまったくトランスペアレントに利用できる.

NonStop 用のディスクドライブは1台約250Mbytes と比較的小型なものを 8 台組み合わせて 1 システムとしており（V 80 ディスクサブシステムの場合), 1 台が致命的損傷を受けた場合の被害を軽減するようになっている. また, アクセスアームが多いので, 同時多発でデータベースにアクセスするようなアプリケーションに適している.

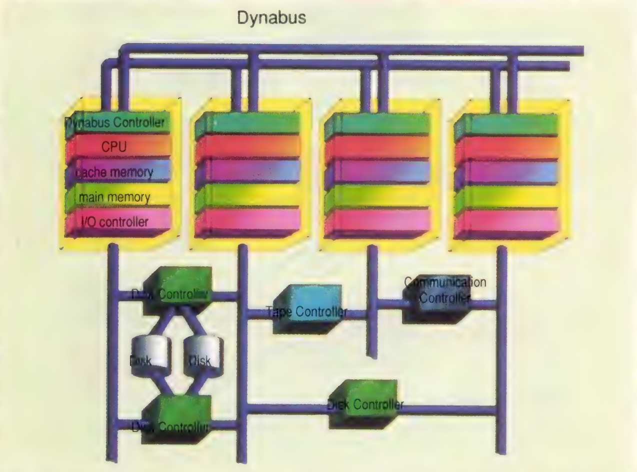


図 1　NonStop ハードウェア

CG/Minako Miyajima

フォールトトレラントプログラム

GUARDIAN90 におけるフォールトトレラントプログラムはスタートアップフェイズとメインループに分かれており, 次のようなシステム特有の処理を行なう必要がある. まず, スタートアップ処理におけるプロセスペアの作成, そして, メインループにおけるチェックポイントの働きについて説明する.

●プロセスペア

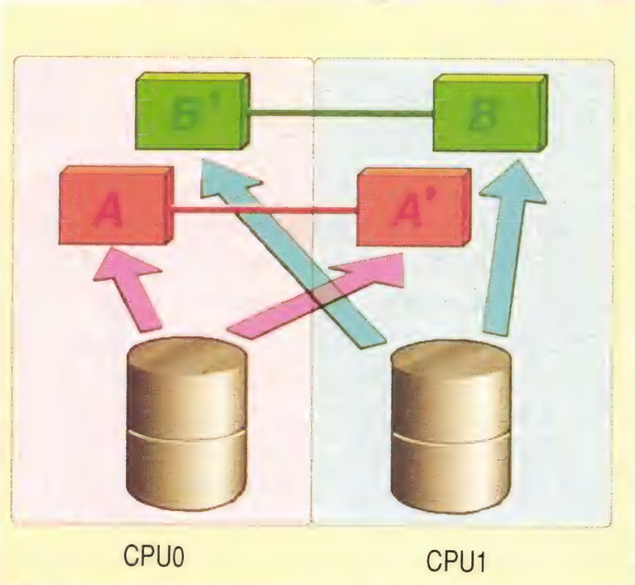


図 2　プロセスペア

GUARDIAN90 はユーザーアプリケーション実行時の耐障害性とデータベースの完全性を実現するため, プロセスペアと呼ばれる概念を採用している. プロセスペアはプライマリプロセスとバックアッププロセスからなり, プライマリプロセスに障害が発生した場合は, すぐさまバックアッププロセスがその処理を継続する.

あるプロセスが実行されるとき, そのプロセスは必要なファイルをオープンした直後, ほかのプロセッサ上に自分とまったく同じ分身であるバックアッププロセスを生成する. そのあと, プライマリプロセスはバックアッププロセスのためのファイルもオープンし, メインループに移る. バックアッププロセスはチェックモニタプロシージャをコールする以外の処理は行なわないが, チェックモニタがプライマリプロセスの障害（プロセッサの障害を含む）を知ることにより, バックアッププロセスはプライマリプロセスの立場に変わる.

スタートアップルーチンで必要な処理を整理すると, プライマリプロセスとして起動された場合は, バックアッププロセスの生成と, プライマリプロセスとバックアッププロセスが必要とするファイルのオープン. バックアッププロセスとして起動された場合はチェックモニタプロシージャをコールすることである. こういった手続きを経たのち, プライマリプロセスはメインループの実行に入らなくてはいけない.

●チェックポイント

プライマリプロセスは，メインループ中でクリティカルな処理（ファイルへの書き込みなど）を実行する前に設定されたチェックポイントで，データをバックアッププロセスへコピーする．この動作はプロセス間通信の形をとって実行される．チェックポイントでコピーされるデータは，プログラムカウンタなどのプロセッサの内部情報，さらにスタック，変数，ファイルバッファ，シンクブロックなどローカルなデータのすべてである．シンクブロックはファイルポインタであるが，それだけでなく最後にチェックポイントを通過してから後のファイルに対する書き込みの履歴（ライトリクエストのIDとその結果）を持っており，バックアッププロセスが発行し直した書き込み要求がプライマリプロセスによって正常終了していなかった場合のみ，ディスクプロセスは要求された書き込みを実行する．

チェックポイントメッセージとして送出されるこれらのデータを保持していることで，バックアッププロセスは最後にチェックポイントを通過した時点から処理を継続でき，データベースの完全性も保証される．処理を引き継いだバックアッププロセスは，自分のバックアップを作成するスタートアップ処理を実行したのち，旧プライマリプロセスの処理を最後に通過したチェックポイントまでさかのぼって実行し始める．すべてのプロセスが常にペアを保ち，チェックポイントメッセージを交換しながら処理を進めるのが，GUARDIAN90におけるフォールトトレラントプログラムの基本的なスタイルといえる．

また，GUARDIAN90ではこのようなプログラム開発の支援のため，TMF（Transaction Monitoring Facility）と呼ばれる環境を提供しており，ALGOL系専用言語TALからTMFのライブラリルーチンを呼ぶことにより，OSの提供するサービスのすべてを利用することができる．

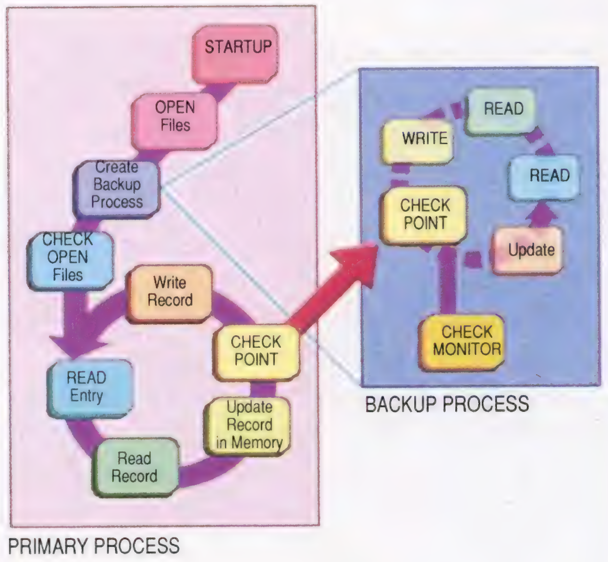


図3　チェックポイント

リカバリとグローバルデータ

●通常のリカバリ

NonStopシステムの各プロセッサモジュールは電源がごく一瞬落ちたような場合もそれを感知して，一定時間続く場合は処理を停止してメモリキープ状態に入る．リカバリの過程は，電源が切れた瞬間，急激に電圧が降下していく瞬間をとらえるところから始まる．

あるレベルまで電圧が下がると割り込みが発生し，プロセッサはレジスタなどすべてのステータスをメモリに退避する．その後ただちにCPUの動作を停止し，それと同時に自動的にメモリバックアップの状態に移行する．そして，電源供給が回復して，あるレベルまで電圧が上がってくるとCPUパワーオンインタラプトが発生し，パワーオンリセットされCPUが立ち上がる．このとき，メモリに退避しておいたステータスを読み込んで，停止直前とまったく同じ状態に復帰する．

まずCPUが最初にもとの状態に戻るが，I／Oの回復はCPUに比べ遅れることになる．特にディスクは回転が安定しアクセスが可能になるまで時間がかかるが，その間CPUはI／Oパワーオンを待つ．障害発生前にI／Oにリクエストしていた場合，I／Oパワーオンと同時にそのI／Oコンプリートのメッセージを受信するが，それが瞬断により失敗していた場合は，再度リクエストを発行して処理を続行する．

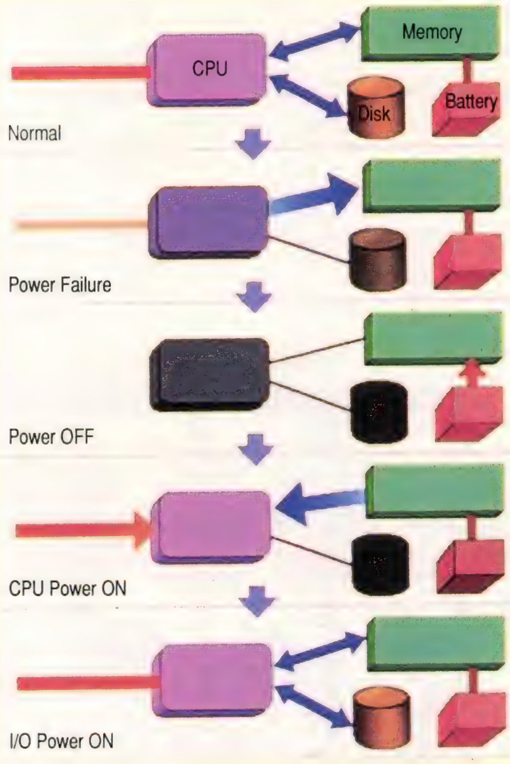


図4　通常のリカバリ

●シンクロナイゼーション

1つのプロセッサが他のプロセッサよりリカバリが遅れたような場合，プロセスペアの管理情報などのグローバルな情報がプロセッサによって異なってしまう可能性がある．特定プロセッサのみのダウンといった状況でも同様の状況が生じる．このようなケースでGUARDIAN90 は，シンクロナイゼーションを守るため，障害を起こしたプロセッサが復帰した後，ただちにそのプロセッサをリセットし，グローバルデータのリロードを行なう．このとき全プロセッサがグローバルアップデートを完了している必要があるので，全プロセッサからのアップデート終了メッセージを受信したのち，今度は全プロセッサに対してグローバルアップデートを禁止して初めて，グローバルデータのリロードが可能となる．

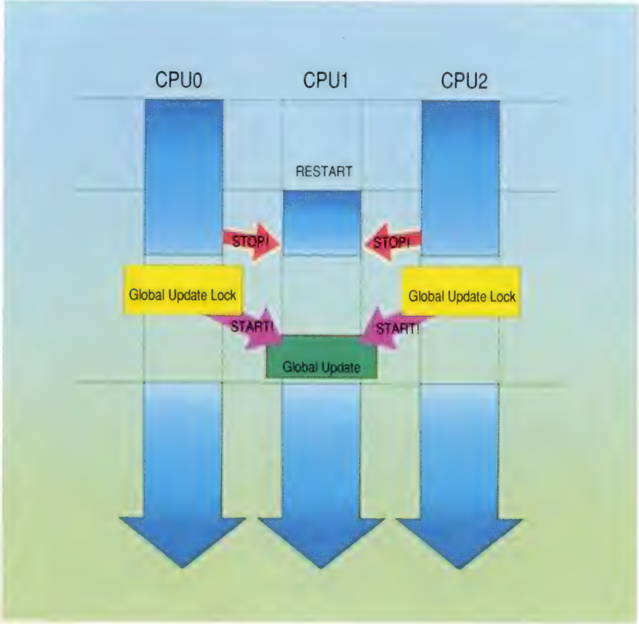


図5　シンクロナイゼーション

●アベレージクロック

疎結合マルチプロセッサ構成でグローバルデータの同一性を保つことは難しい．GUARDIAN90 で重要なグローバル情報はプロセスペアのテーブルのほかにシステムの時刻が挙げられる．各プロセッサは独立したクロックで動作しているので，それぞれが示す時刻には違いが生じてしまう．システム時刻がラジオの時報と一致していることはそう重要ではないが，各プロセッサが異なったタイムスタンプを発行してしまうと，プロセッサなどの異状と誤まられて問題が大きい．そこで NonStop システムでは各プロセッサの平均クロックを標準として，それぞれのプロセッサがそれに自分の時計を合わせている．

時刻調整はシステム外部から時刻を入力してやることで可能だが，120秒以内の誤差の場合，各プロセッサは与えられた数値に近付くようにクロックの進みを0.1秒単位で調整していく．この場合もプロセッサ間のズレが生じないように同期をとりながら目標の値にゆるやかに時刻を変化させていく．

NonStop 多重端末制御

● PATHWAY

大規模なOLTP業務になると，1CPU当たり1,000端末以上の処理を行なうこともある．この場合，プロセスの多重スレッド化なしでは，プロセス切り替えのオーバーヘッドが大きくなりパフォーマンスが低下する．このようなユーザーアプリケーションの開発は難度が高いが，NonStop システムで提供される PATHWAY トランザクション処理システムには，フォールトトレラントに動作する TCP(Terminal Control Process)をはじめとする多重端末制御環境が整備されている．

PATHWAY の TCP は OS のカーネルよりもはるかに大規模で複雑なものだが，TCP レベルでフェイルセーフが保証されていることにより，スクリーン COBOL と呼ばれるインタープリタによって端末のスクリーンレイアウト等を記述するだけで，銀行のATMシステムクラスのOLTPにも対応できる．PATHWAY により提供される環境では不十分な用途では，TCP に相当する部分を TAL で記述することにより，フォールトトレラントなシステムを構築する必要がある．しかし，PATHWAY はもともと OLTP 向けのソフトウェアであるのでその必要性はあまりなく，ユーザーレベルでは TAL は回線まわりのスループット向上といった限られた用途に使われる．

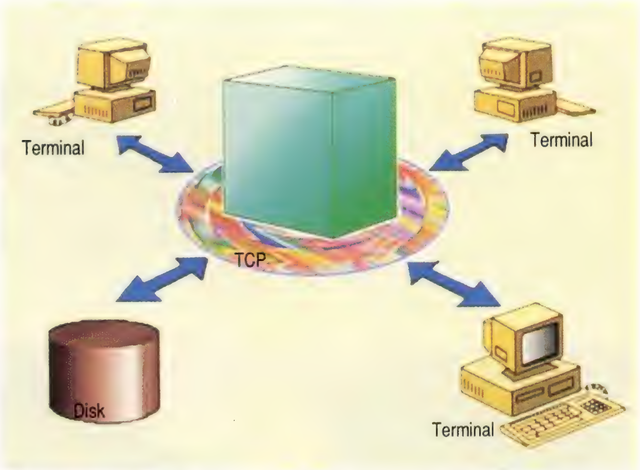
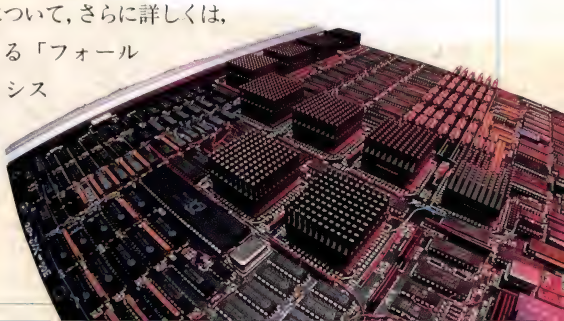


図6　PATHWAYとTCP

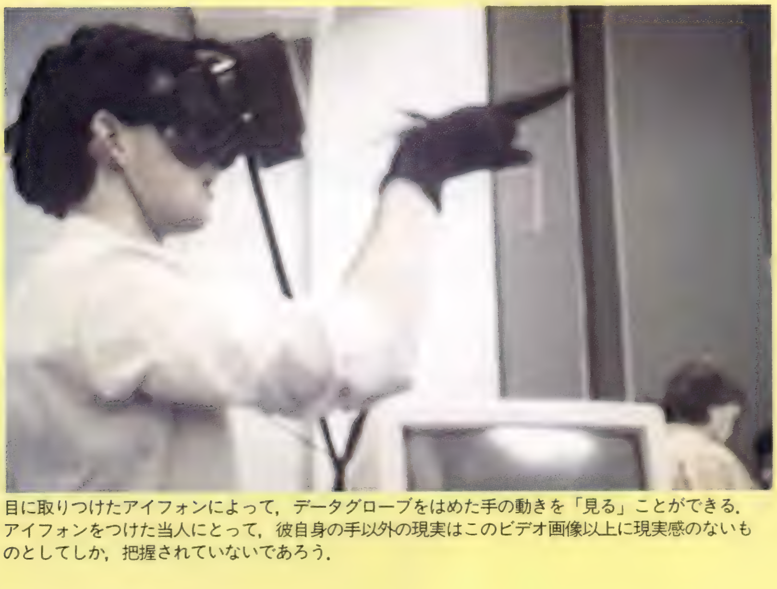
まとめ

NonStopでは，メインフレームで一般的なホットスタンバイによらず，高度な OS が要求される疎結合マルチプロセッサというアーキテクチャによって，フォールトトレランスと OLTP 分野での高いコストパフォーマンスを両立させているといえる．NonStopの実装について，さらに詳しくは，J.グレイなどによる「フォールト・トレラント・システム」を読むことをおすすめしておこう．



VR
VIRTUAL REALITY

ヴァーチャル・リアリティとは何か？　この疑問に3つの立場から並行して答えていく．仮想現実と現実とが並行して存在する状況の持つ意味を，記事構成からも感じてもらいたい．さらにヴァーチャル・リアリティの作り手たちへのインタビューも加えた．ヴァーチャル・リアリティの現実度を評価するための材料として，十分なはずである．



目に取りつけたアイフォンによって，データグローブをはめた手の動きを「見る」ことができる．アイフォンをつけた当人にとって，彼自身の手以外の現実はこのビデオ画像以上に現実感のないものとしてしか，把握されていないであろう．

データスーツやデータグローブからの人の挙動のデータが，グラフィックワークステーションで処理され，アイフォンにヴァーチャル・リアリティでの出来事としてリアルタイムで立体表示されていく．

複数の現実とテクノ・リアリティ ヴァーチャル・リアリティの現在

武邑光裕

人工現実(Artificial Reality)の実現に重要な基礎となってきたシミュレーション技術の進化は，インタラクティブCGやマルチ・メディアの熟成に伴い，VR——ヴァーチャル・リアリティ（Virtual Reality=仮想現実）による「想像上の現実感の生成」を促し，いわば電子メディア環境が構成する「現実の複数化」という刺激的な状況を生み出しつつあるのは周知の通りである．

CGに限らず，コンピュータとアプリケーションの生みの親ともいえるアイヴァン・サザーランドは，かつてコンピュータを「アリスの不思議の国」にたとえ，一方では忠実に現実を模倣したり再現したりする目的を持つCGは，もう一方ではまったく現実には存在しない抽象的で想像的世界をモデルとして描くことができる最良の表現手段であると指摘した．この不思議の国のデザイニングこそ，現在最も注目されているVRの大きな可能性であり，夢や幻覚を含む脳内の有機的な情報を立体化する有効な手段ともいえる．

VPLリサーチ社を創設した若き天才プ

今年，テキサスで行なわれたサイバースペースに関する国際会議では，エリック・グリッフセンのシステムなど刺激的な展示がみられた．

サイバースペースの幽霊 もしくは，現実社会の崩壊

伊藤ガビン

ほんの1年前には，よほどコンピュータカルチャーに詳しい者か，あるいは限られた技術者の間でしか聞くことのなかったヴァーチャル・リアリティという言葉が，ここ半年程度の間に急激に普及した．一般誌でも，文化誌でも，テレビでまで取り上げられ，ある種ブームのようになっている．

実際，ヴァーチャル・リアリティという概念はかなり魅力的であるし，データグローブなどのアイデアは分かりやすく，またコンピュータは苦手という頑固な文系人間にもとっつきやすい存在だろう．そして，もちろん斬新でもある．

しかし，道具や技術というものが，すべて人間の機能を拡大するものであるとするならば，ヴァーチャル・リアリティの研究もあって当然といおうか，非常に正統的なテクノロジーの進化史の中に位置付けることができる．ただ，その研究の視線が，人間の外部器官の拡張だけではなく，内部，あるいは精神の領域にまで及んでいる点が美しい．

SF作家であり，コンピュータ・サイエンティストでもあるルディ・ラッカーは，

自転車をこぐにつれて，風景が変質していく．一般的なドライブゲームと本質的に同じものだが，ヴァーチャル・リアリティの一例でもある．

ヴァーチャル・リアリティの 原材料と問題点を探る

編集部

グラフィックワークステーションの高性能化は，リアルタイム3Dアニメーションという新しい技術を生み出した．とはいえ，これは単にコンピュータの速度の問題にすぎない．無数のポリゴンによって面を分割し，色を与え，質感を与え，光を与えることによって，真にフォト・リアリスティックな画像を作りだすことができるならば，ポリゴンの位置の変化を与えることによって動画を手にするまでに，立ちはだかる技術的障壁は何もなかった．

しかし，現実世界の一瞬をポリゴンの座標に縮退させるのは職人芸によってなしうることではあるけれど，時間の流れによる現実の変化をポリゴンによって作られた世界に取り込むには，工夫をしてやらねばならない．2次元の静止画の場合にデジタイザを使って座標を抽出したように，空間座標とその変化を読み込むハードウェアを用意することになる．

1点の情報でよいのであれば，磁気センサによって物の位置座標をリアルタイムに取り込むのもさほど難しくはない．しかし，現実の物体を微小な面に分割し，その各頂点にそのようなセンサを取り付け，そのまま画像を作るというアプローチには物量的に無理がある．そのような力任せのアプローチが通用するのは，コンピュータの中だけだ．

VPL社のデータスーツは，人間にウェットスーツのようなものを着せ，そのスーツの要所要所の曲がり具合を取り込むことによって，人間の動きを取り込もうとするシステムである．人間の解剖学的構造を利用すれば，比較的少ないデータ量で人間の姿勢や動きを把握できるわけだ．これによって，現実世界のうちから

ログラマであるジャロン・ラニアは，VRを人間の欲望や夢を吸い取る巨大なスポンジであるとたとえたが，私たちは，VRの内部に，いまだかつて体験したことのない創造的現実の可能性を発見しつつ，そのリアリティーを一定の現実にフィードバックさせるという回路を所有することになる．こうしたいわば複数の現実を階層的に，あるいはフュージョン化するプロセスは，より人間の知覚器官や神経系などの有機的な身体性と，外部メディアとの関係回路を立体化する生体工学と情報との交流を活性化させていく視点ともいえるだろう．

こうしたVRのスリリングな可能性の中で，いくつかの実際的な動向についてレポートしてみよう．まず，現在アメリカ指導による中心的なVRプロジェクトには，ジャロン・ラニアのVPLリサーチ社をはじめ，NASAのエイムズ研究センターで，仮想環境表示システムとして知られる遠隔通信ロボット制御技術の開発に関与し，最近NASAを辞めて動向が注目されるスコット・フィッシャーの研究が挙げられる．MITメディア・ラボで，高等視覚研究やステレオ・グラフィックスの可能性を探究してきた経過からVR，特にテレイグジスタンスとして注目を集める遠隔通信制御や仮想環境再現技術における彼の業績は，きわめて大きなものがある．

さらに，CAD／CAMの先端ソフト開発で知られるオートデスク社のサイバースペース・プロジェクトのランデル・ウォルサーは，現在のオートデスク社におけるVRプログラムの中心的存在として，建築からゲームまでの幅広いVRアプリケーションの開発を遂行している．そして，VRの先端技術開発のメインキャストの中でも，最も斬新かつ活動的なプログラマが，先のオートデスク社を離れ，PCレベルでのVR応用に一歩先んじる展開を開始

未来を思い描くことができなくなったといわれるのは，月世界旅行などのイメージを消費しつくしただけであって，今後の未来像は人間の内的世界や数学的世界の中にあるのではないかというような意味のことを言っていた．まさに内的世界は，地球上のいまだ開拓されぬ新たなフロンティアとして設定され，電子テクノロジーの力によって原石を見つけ出し，研かれようとしているのだ．

精神の領域に関する機能の拡張，つまり意識拡張という概念は，ヒッピーカルチャーやドラッグカルチャーを代表する重要なものの1つだ．データグローブの開発者ジャロン・ラニアの風貌を見るまでもなく，ヴァーチャル・リアリティ研究を牽引しているのは，かつてのヒッピーカルチャー，ドラッグカルチャーの影響下にある人物たちなのである．

なぜ彼らが，そのままドラッグの世界に閉じこもらないで電子メディアの世界にやってきたのかは分からない．しかし，ユートピア建設にしろ，インナービジョンの他人との共有にしろ，コンピュータ内でのほうが，ずっと簡単にできることは確かなのである．まあ，彼らがやりたいことはそう単純なことではないのだが．

それでは，コンピュータによる意識拡

人が着てデータスーツに動きを供給することによって，演じ手へのフィードバックのない単なるアニメーションを作ることもできる．

テレビドラマ，ネットワークベイビーのネットワークゲームに登場するキャラクターの操作にも，データグローブが使用された．

1人の人間の動きを実時間で取り出せる．この結果，現実の人間の動きを別の人間の姿と組み合わせた画像を作り出すことも可能になる．

データスーツの機能は，現実から非現実への座標変換である．データスーツは，そういう意味において，現実の人間あるいは現実自体と深い機能的な関係を持っていることになる．しかしデータスーツは，本来この逆の変換を行なうために作られたデバイスである．まず人間に仮想的環境を与え，その環境の中での動作をデータスーツによって引きずり出す．この動作を現実世界にあるコンピュータによって処理し，仮想的環境にフィードバックをかける．お分かりのことと思うが，最初の仮想的環境もコンピュータを介して与えられるものである．

VPLのシステムで仮想的環境を提示するための物理的手段として使われているのは，両眼のそれぞれに3Dグラフィックスワークステーションのディスプレイ（古典的な仮想的環境提供の手段であるearphoneに対抗して，eyephoneと呼ばれている）で覆い，立体視させるシステムである．このシステムでは，ディスプレイによって現実への視線は完全に遮られ，コンピュータがデータスーツを着た人物の

したセンス・8（エイト）社を創設したエリック・グリッフセンとパトライス・ゲルバンドである．

特にエリックとパットによる低価格VRの開発スピードはめざましいものがあり，システム価格4000ドルのSun　SPARC stationを用いた彼らのVRシステムは，2台のシリコン・グラフィックス・ワークステーションを用いた通常のVR再現と比較して，解像度では劣るものの，リアルタイム・スピード再現においてはきわめて高性能なインタラクションを実現している．

今年の4月下旬，テキサス大学で開かれた「サイバースペース国際会議」では，SF作家のブルース・スターリングとエリックによるVRフィクション（VRと物語性）に関する論議を期待したが，エリックの開発したPC上のVRトラベリング・システムが大きな話題となり，筆者自身，スターリングと共に，エリックの方向性に対してきわめて強い刺激を受けた．さらにエリックは，この夏にもAmiga3000に6種のカスタム・チップを供給し，グラフィックス，ヴィデオ，サウンドを軸にしたVRのリアルタイム・マルチメディア・プロダクションが可能となる統合型ソフトウェアの発表を目前としている．ちなみに，このシステムの全貌は，9月の段階で明らかになるだろう．

こうしたPCレベルでのVRソフトの開発は，現在把捉されているだけでも相当数のプロジェクトがあり，今後ゲーム・エンターテイメントへの波及は時間の問題でもある．中でも前述のエリック・グリッフセンのアプローチに次いで，カリフォルニアのVRL(Virtual Reality Laboratories)社が発表したAmiga用プログラム“Vista”は，先に発表になったフラクタル無限階層へのトラベリング・ソフト，“Fractal Flights”を延長した太陽系探査プログラムであり，いずれ一般化するで

張，ヴァーチャル・リアリティの世界における意識拡張とはどんなものだろう？現在のコンピュータネットワークの一歩進んだ形で，インターフェイス（そのときはもはやインターフェイスと呼べないかもしれないが）が画期的に変化したシステムの上で起こる意識の変化である．

それは，1つには時間に対する意識の変化であろう，そして距離，大きさ，つまりあらゆるスケールに関する意識の変化だ．ヴァーチャル・リアリティの世界ではスケールに何の意味も持たすことはできない．時間は流れないし，大きさはすべて相対的なものだし，距離は誰かが設定しない限り，ない．そして，情報に関する意識．あるいは土地やお金．つまり所有という概念の無効化だ．コンピュータの中で何かを所有しても，それは，単にデジタルな情報が書き換えられただけで，本質的に何の意味も持たない．

これらの意識の変化は，現在でもネットワークに没入しているときや，使いやすいように整理されたデータの沃野をさまよっていると，かすかに味わうことができる．たとえば，瞬時にあらゆるデータにアクセスできる環境に浸っていて，突然秋葉原に物を探しに行ったりすると，この現実のアクセススピードの遅さにあきれてしまうことがないだろうか．

こうしたコンピュータ内環境は，今のところキーボードやマウスといった特殊なインターフェイス（みなさんは好きかもしれないが）を介し，しかもリアルタイムでやり取りされるデータがテキストだけなので，まだ現実との区別をつけることができるが，完全に生体情報を転写できるインターフェイスが確立され，ましてや脳からダイレクトに情報が送り込めるようになると，もう現実に何らの意味や価値を見い出せなくなるかもしれないのだ．

しかし，そうなったときに，つまり現

視覚をすべて制御することになる．

この人物の視野はあくまで1人称であって，データスーツの屈伸と移動，すなわちデータスーツを着た人物の動作によって，現実世界における視野の変更と同じように変化する．また，現実においてそうであるようにデータスーツを着た人物自身も視野に入る．視覚上はすべての動作について，現実世界におけるのと同様のフィードバックが返ってくることになる．

このような技術を使えば，現実の空間での動作と人間が入っていけない危険な場所にいるロボットの動作，さらにデータ上でしか存在しない場所での架空の動作とをすべて同じように感じ，行動することができる．仮想的環境のデータが，現実そのものから取られているか，あるいはロボットのセンサから取られているか，また純粋なコンピュータ内部のデータとして取られているかというだけの違いである．また架空の動作は現実のシミュレーションであってもよいのだが，現実とはまったく無縁の環境におけるものであってもよい．

また，このようなデバイスを付けた人物2人を同一の環境に置くことができる．これはつまり現実とは無縁の環境を，2

データスーツを使うには，実際に使う人間に合わせて，事前に動きのキャリブレーションをしてやる必要がある．

あろう「ヘッド・マウンテッド・ディスプレイ」や，無限遠近ディスプレイ・システムとの連結によって，一種のアストラル・トラベルが体感上で実現されるというソフトである．

一方，PCレベルでのVR開発と対極にあるのが，NASAやDARPA（The Defense Advanced Research Projects Agency）によるスパコン利用による大型VR再現であり，ここでは地球から火星，ウラヌスやミランダを越えて自在にフライトが可能となるシミュレータが完成している．軍事利用におけるフライト・シミュレータでのVR再現はもとより，最近の映画「レッド・オクトーバーを追え」で登場した海底地形のVR再現システムなどは，現実に人工現実再現の先鋭技術として注目を集めており，いずれ全地球を含む多彩なVRシミュレータの情報宇宙は，飛躍的な進化を遂げることは疑いえない．

ジャロン・ラニアとVPL社の設定するVR用シリコン・グラフィックス・システムが約10万ドルであることを考慮するなら，Amiga3000上を疾駆する完成度の高いVRシステムは，約2万ドルと想定されている．先のVRL社がAmigaのVR向けにターゲットを絞って開発中のプログラム“Distant Suns”は，紀元前8000年前までさかのぼって全地球を視野の中に置くもので，今後AmigaでのVRプログラム・サプライは刺激的な状況を喚起していくことは間違いないだろう．

こうした状況にあって，’90年代におけるVRの多彩なアプリケーションを支える周辺機器のイノヴェーションも急速に開発が進められており，ボストンのリープ・システム社によるヘッド・マウンテッド・ディスプレイ開発や，その簡易システムである「リープ・システム」の発売などは今後のゲーム・エンターテイメント・デヴァイスのプロトモデルとして注目を集めている．今後，日本の家電メーカー

実よりもヴァーチャルな現実のリアリティが増したときに，人間には果たして，まだ肉体を捨てずに生き続けることが必要なのか，生き続けることが可能なんだろうか？　サイバーパンク小説のように，生命のアルゴリズムを発見し，ネットワーク内を幽霊のように漂うことができるのか？　そう，それは幽霊のように死んだ人間の残りカスなのか，あるいは不老長寿の完成なのだろうか？　まあ，そんな話は今考えたところで結論が出るものでもない．とりあえずは，現実というものが，スケールや所有という物理的条件（まさに「物理」的だ）にいかに制約を受けているかに注目する必要がある．

さて，その現実の制約だが，ここまで述べてきたのは主に意識に関する制約の部分だったが，これと同程度に重要なのが身体機能の制約だろう．たとえば目の制約，目の解像度，目のシステムだ．

普段，何気なく生活していると気が付かないが，私たちの思考はかなりの部分，目というインターフェイスに依存している．脳内プロセッシング能力とは関係なく，創造できる色は目で見ることができる色だし，眠っているときは目など使っていないにもかかわらず，見る夢は左右に配置されたカメラから入力された立体視映像である．決して360度全周囲を一度に把握した映像ではない．これは見るという行為そのものが，左右カメラの立体視映像だとプログラムされているためだが，脳のポテンシャルを考えると，この程度しかできないわけはない．単に目のシステムに脳が馴れているだけだろう．

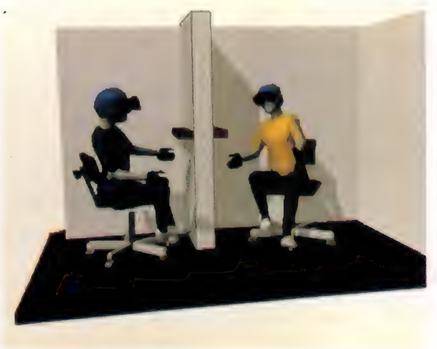
そこで，従来使用していた（今も使ってるけど）目のシステムを見直して，別の“見る”ためのシステムを考案することになる．ラピス・テクノロジー社のステファン・ベックが開発中のホスフォトロンも，こうしたアプローチで作られた網膜劇場デバイスである．このシステムで

人の人間が共有することができるということである．このような環境を多くの人間が共有すれば，新たな社会が生まれる．この社会はInternetなどに代表される現在のネットワーク社会以上のダイナミックさと，国家あるいは地域といった現実の社会構造以上のリアリティーを持って存在することになる．もちろん物理的な距離は，この社会の広がりにとって妨げにならない．

このようなヴァーチャル・リアリティを作り出すための現実，すなわちグラフィックハードウェアやネットワークはすでに存在している．ヴァーチャル・リアリティの現実世界における実現は，ひとえに時間の問題である．

ただしヴァーチャル・リアリティにおける時間の問題は，決して単なる時間の問題ではない．ヴァーチャル・リアリティのきわめて初期の実現ともいえる，人工衛星のテレメータによる制御などで，すでに1950年代から問題になっていたように，手を動かしてから手が動くのが見えるまでの時間が問題なのだ．

遠隔地にあるセンサの場合と同じように，あるいはそれ以上にヴァーチャル・リアリティにおいて，リアルタイムのレスポンスを与えることは難しい．同じ環境にアクセスする人間あるいはコンピュー



壁によって隔てられた人どうしでも，アイフォンとデータスーツを使って，物のやり取りやダイレクトなコミュニケーションをすることができる．

からゲーム・メーカー，さらにはゲームハードウェア・メーカーにとっても，VRとその周辺領域の拡大は，予想以上のメディア・フュージョン化現象を伴って拡大していくと思われることから，これらのすう勢からはここ当分の間目が離せないであろう．

エリック・グリッフセンが開発した新しい立体視システムは，フライングチェアという椅子と組み合わされ，体験者にヴァーチャル・リアリティの無重力感覚を与える．

最後に，ジャロン・ラニアが年内にスタートさせるVRネットワークについて簡単に紹介しておこう．これはVPLと3つのVR研究機関（ノースカロライナ大学，メディア・ラボ，ワシントン大学）を結ぶ“Reality Net”と呼ぶ世界初のVRネットであり，今後，AT&Tやパシフィック・ベル社の特別電話回線を通じて実験がスタートする．このようにVRトラベラーの実現を目前にした急激なVR展開が進んでいる．国際会議に関しても，夏以降年内の主だったものだけでも数種の会議がスケジューリングされており，今後もVR元年をスプリング・ボードとした急激な展開が予測されている．

エンターテイメント，教育，通信，サイバーメディア・フュージョン，インタラクティブ・シミュレーション，医学等々，VRをめぐる複数の現実に向けられた視点移動こそ，仮想現実の生成を超えた電子情報生態系の未来を見通す先鋭的なまなざしなのである．

は，映像は光の波長として入力されるのではない．特殊なゴーグルの横，ちょうどコメカミに付けられた電極から流れる微弱電流の効果によって網膜にある刺激が伝わり，それが光として脳に伝達されるという仕組みらしい．いわば，生体内の情報処理プロセスを分析し，そのシステムを使って別情報をダイレクトに脳に伝える実験である．

こうした試みには，音の部分でも3次元立体音像が得られるホロフォニックシステムなどがある．これら生体内の情報処理プロセスの分析も，もの凄いスピードで進んでおり，その情報を直接脳からピックアップしたり，逆に脳に情報を送り込んだりする実験も始まっている．いわゆるウメコミとかジャックインだ．

もう，こうなってしまえば，何の意識的操作なしに，情報が流通するようになる．それはマンマシンインターフェイスではなく，マンマシンそのものだ．人類のうちの果たして何人がマンマシンになりたいと思うかは分からないけれど，とりあえず私はなりたいと思う．そしてネットワークでつながって，ネットワークという名の巨大な生物の一部となってもいいと思う．それが幸せかどうかは知らないけれど．

リーブ・システムは，自分の目を取り外してしまい，手で持って動かすことによってヴァーチャル・リアリティを達成できる．

タたちによってなしうることの総合計は，容易にコンピュータの処理能力を上回るのである．

その結果，注意を引いたもののほうに顔を向けるというような，ごく単純な行為すらできにくくなる．リアルタイムのフィードバックなしには，人間は自分の顔の向きのコントロールさえできない．当然，物をつかむといった全面的にフィードバックに依存した行為は，かなりの熟練を必要とすることになる．

このようにして，自分の行為が，ヴァーチャル・リアリティに伝わって戻ってくるまでのディレイが生じ，この結果ヴァーチャル・リアリティとそこに入っていこうとする人間の間に距離が生まれる．この空間的歪みを物理的に解消するための技術は，まだ確立されていない．今後のコンピュータテクノロジーの進歩に依存する問題であろう．

しかし，この問題を解消するには，テクノロジーの発達を待たなければならないというものでもない．この歪みを心理的に解消するというアプローチがありうる．すなわち時間的ディレイをある程度の範囲において，人間が自力で補正するのである．これは，人間の適応力を考えると不可能なことではない．

この補正は，さらにおもしろい側面を持っている．補正が日常的になることによって，人間はヴァーチャル・リアリティに適応し，ヴァーチャル・リアリティは，さらに超現実的存在となる．補正に慣れた人間は，無重力に慣れた人間と同様に，この地上に戻ってくるのが難しくなるのだ．

このような歪みの補正に慣れた人間たちが集まるヴァーチャル・リアリティこそが，まさに仮想現実の名にふさわしい継続性といびつさを持った世界になるであろう．このような人間たちの変化のためにも時間は必要なのである．

アイフォンとデータスーツ 安定と変革
Jaron Lanier

Lanierは尺八を買いに出て，少し遅れて戻ってきた．

——どうして，データスーツとかアイフォンとかを作り始められたのですか？

Lanier：人間の感覚のシミュレーションがやりたかったからです．天使や幽霊のように，どこにでも入り込めますしね．また，夢と会話の混ざったようなコミュニケーションの道具でもあります．そしてもちろん，多くの人が我々の製品を買ってくれたからでもありますね(笑)．

——いつかみんなが，データグローブやデータスーツを着けて歩くようになるでしょうか？

Lanier：ファミコンのパワーグローブは，その先駆けですね．軽くてワイヤレスのデバイスが必要ですし，ゲームマシン並みに安くて，ワークステーション並みに強力なコンピュータが必要です．あとはどれだけの能力が必要かの判断ですね．

——みんなが常時仮想世界に行けるような社会では，いったいどんなことが起きるでしょうか？

Lanier：仮想世界では，どこにでもいつでも行けますし，ありとあらゆるものを一瞬で作れます．アナーキーな世界でもあるでしょうし，同時に非常に厳格な世界でもあることでしょう．また，そんな世界では変化が当たり前で，3日も同じ建物が建っていたら，驚異的です．それでは，変化そのものが退屈になってしまいませんか？　それにひきかえ現実世界では，樹木などをゆっくり時間をかけて育てていくことができます．仮想世界ではそんなことはできません．仮想世界にいる時間が長くなればなるほど，現実の世界が恋しくなるでしょうね．

——そんな世界に行ったきりになる人も出ると思いますが？

Lanier：そんな人は戻ってこないほうが現実世界が落ち着いたものになって，我々のためにいいかもしれませんね．私は必ず現実に戻ってくるつもりです．

アイフォンやデータスーツなどのシステムを作り出したVPL社の社長，音楽家でもあるJaron Lanier.

INTERVIEW

5月30日に東京のTEPIAで開かれた国際シンポジウム，「サイバースペース虚像現実の創造」での発表のため，ヴァーチャル・リアリティのリーダーたちが来日した．その機会を利用してのインタビューである．

ウェットウェア，ソフトウェアなどの作品を生み出したSF作家にしてMathenaut, Rudy Rucker.

セルラーオートマトンと数学 ヴァーチャル・リアリティとSF
Rudy Rucker

Ruckerはいきなりウイスキーを注文して，インタビューにこたえ始めた．

——ヴァーチャル・リアリティをどう思われますか？

Rucker：エキサイティングだね．まず，マンデルブロのような理論的なものの観察ができる．それと，ロボットのような人工の生き物のテストをするのによい場所だ．ロボットを作るのは難しいことだけれどね．また，ヴァーチャル・リアリティを共有すると，テレビやビデオにはできなかったようなコミュニケーションの道具として使える．3Dの環境は，コンピュータを使うためのインターフェイスとしてもおもしろい．ウィンドウシステムを超えるものが作れると思うよ．

——数学的なコンセプトの表現との関連はどうでしょう？

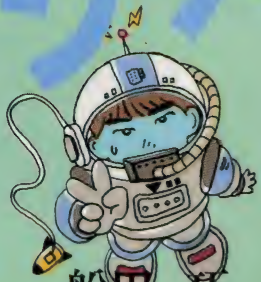
Rucker：ワークステーションレベルのグラフィックスと組み合わせると相当なことができる．タイタンワークステーションのデモで，旗が風になびく様子のアニメーションがあるんだが，数学的に単純なモデルなのに，動きは複雑で興味深いものだ．ほかにも，ハイパーキューブの交点とか，カタストロフのようにダイナミックなシチュエーションとか，科学的な可視化のツールとして見るとおもしろいね．

最先端

システム探検隊

〈連載2〉

国立天文台の巻



船田 巧

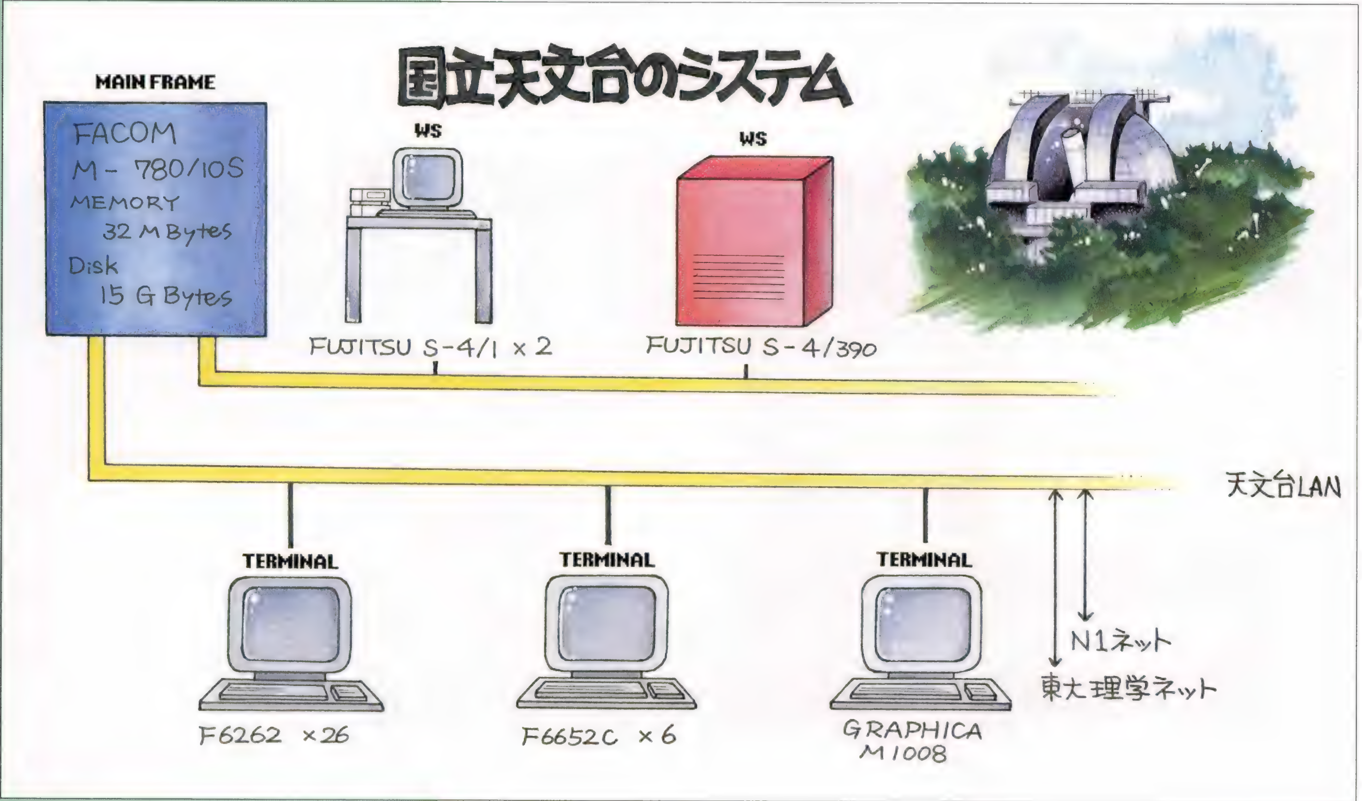
壮大かつ深遠，そしてロマンチック．それが天文学だ．コンピュータシステムの最先端度という点では今一歩だったけど，これから新しいマシンも導入されて，先端的な利用も増えていきそうな気配が感じられたのであった．

10m望遠鏡．世界で10番目くらいに大きい．古い設備だが，まだまだ現役．

William Poundstoneの『ライフゲイムの宇宙』という本を読んだんだけど，これがなかなかおもしろかったのだ．数十ステップのアセンブラで記述できるライフゲームの宇宙が自己増殖機械を生み出すこともできるという話と，最大の複雑さを持つ実際の宇宙がどのようにして生まれたかを解き明かそうとする科学者たちの話が交互に進行して，最後に，宇宙もライフゲームと同様，ある法則によって再帰的に定義された存在なのだと結論付ける．

では，宇宙を記述する法則とはどんなものなのかというと，それはまだ不完全にしか分かっていない．宇宙のアルゴリズムを見つけようとしている人たちは物理学，数学，情報科学などたくさんの分野にいるが，宇宙の研究を本業としているのが天文学の研究者たちである．

あ，また前フリが長くなってしまったが，というわけで今回は東京三鷹の国立天文台に行ってきたのだ．日本の天文学の総本山である国立天文台は，武蔵野の面影残る緑に囲まれた美しい研究施設である．探検隊のお相手をしてくださったのは，天文学データ解析計算センター長



富士通S-4（Sun4のOEM機）の前に座る西村史郎国立天文台教授・天文学データ解析計算センター長．長年，観測と計算機応用の接点を務めてきた．

国立天文台のメインフレームは富士通M-780．ブートは毎朝8時30分，停止は午後7時30分だが，ジョブが多い日は終夜運転される．

の西村史郎さん．この方は日本の天文学における計算機利用の草分けのひとりで，初めて光学望遠鏡とパラメトロン計算機を使い，星からくる光の波長を計測したのが，20年以上前のことである．

「そのパラメトロン計算機のメモリ容量は512ワードでした．1ワードは50bitです．アセンブラが使えましたね．観測データの入力は，ペーパーレコーダに出力したスペクトラム線の位置をものさしで測って，それを自分でパンチしました．パラメトロンの次はゲルマニウムトランジスタを採用した沖電気製のコンピュータで，星の明るさの測定をしました．光学望遠鏡に光電管を取り付け，その電流をAD変換して紙テープに出力したものを計算機に読ませて，そのデータを積分してやるわけです」

明度を算出するには，大気の影響などを補正するために，数十次元のマトリックスを扱う最小2乗法による計算をする必要がある．電子計算機を使い始めるまでは，タイガー計算機（1923年に大本寅次郎が開発した手回し式計算機）で計算していたとのこと．

ところで，天文とコンピュータといえば，1970年頃アメリカ国立電波天文台で実用化されたプログラミング言語FORTHの名が思い浮かぶけど，日本ではあまり使われなかったようだ．

「1973年にはミニコンで望遠鏡の機械制御を始めました．そのシステムは時間分解能1ミリ秒で，主にパルサーの観測に使われました．望遠鏡の制御にはFORTHではなく，FORTRANが使われていましたね」

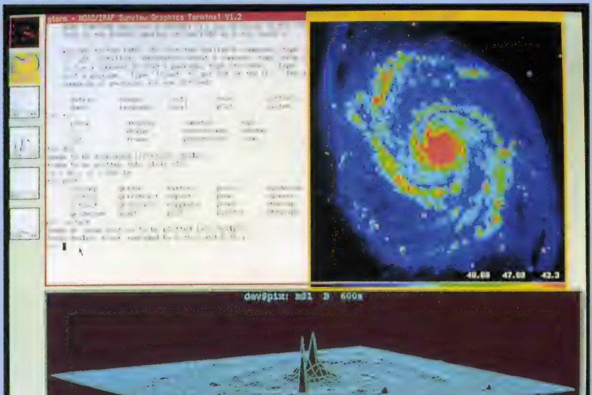
ガランとした端末室．涼しい日はみんな自分の部屋の端末からログインするらしい．

現在，西村さんは天文データベースの構築と運用も手掛けている．天文の分野ではかなり早い時期から星のデータベース作りが始められていて，今のところ約50万個の星について，その名前や位置，明るさなどが検索できるようになっている．

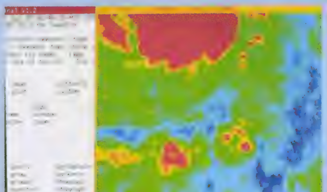
西村さんの言い方を借りると，天文学は博物学的で，植物学や地質学のように

画像データ解析ソフト“IRAF”

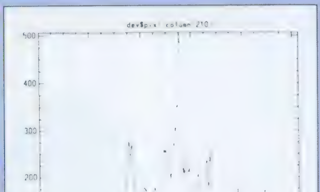
IRAF（Image Reduction Analysis Facility）はアメリカ国立天文台で開発された，対話型の画像データ解析ソフトウェアである．簡単なコマンドで画像データ間の演算，グラフ化をはじめとする各種処理が可能である．SPPというCに似た専用言語で記述されており，SPPを使うことにより機能を拡張することができる．また，既存のソフトウェアを生かすためのFORTRAN to SPPコンバータも用意されている



最大512×512ピクセルのイメージを処理することができる．Ver.2.8はSunView上で稼働しており，X-Window用のVer.2.9も完成している．



猟犬座M51星雲の一部を拡大．拡大，色変換などはマウスで指示できる．



星雲の明るさを断面図状にグラフ化したもの．アウトライン抽出の機能も持つ．

たくさんのサンプルを集め，その中から変わった事物を探し出す学問といえる．だから，50万の星が入ったデータベースが意味を持つ．

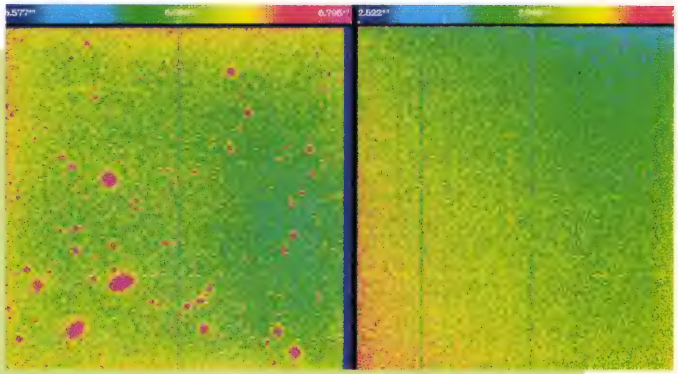
この50万の星々が天文学の歴史を物語ってもいるわけだが，ここ1年の間に位置や明るさが分かっている星は急速に増え，50万から一挙に1400万になった．この歴史的大躍進はアメリカが打ち上げた天体観測衛星“ハッブルスペーステレスコープ”のおかげである．

大気圏の外側で観測するスペーステレスコープは，15等級程度の暗い星までCCDで撮影することができる．1枚の画像データは数Mbytesの大きさで，それがなんと年間1Tbytes分送られてくるのだ．そこで，この膨大なデータを運用できる，大規模なデータベースのインフラ作りが急務となってきた．

今のところ，国際的な天文のデータベースセンターはアメリカ，フランス，東ドイツ，ソ連，そして日本(国立天文台)に存在しているが，日本はアジア諸国にデータを供給する担当国でもあるので，責任がある．

CPUパワーの増強も国立天文台の課題に挙げられている．以前からスーパーコンピュータの購入が検討されてきたが，日米貿易摩擦の影響で遅れていた．どうやら，現時点における国立天文台の計算機環境は十分とはいえないようだ．共同利用機関という性格上，さまざまな要求に応じられる計算機資源の整備が求められる．幸い，スーパーコンピュータは来年度から予算申請できるらしい．

国立天文台の施設である野辺山観測所では，スーパーコンピュータで観測データの解析を行なっている．



IRAFのほかに，長野県の木曽観測所で開発されたSPIRALという画像処理ソフトも使われている．上はその出力例．

SPIRALはFORTRANで記述されている．M-780上で稼働するので，ワークステーション単体より処理は速い．一見旧式な端末もけっこう使いやすそうである．

では，スーパーコンピュータの使い道としてはどんなものが考えられているのだろうか．

「観測データを大計算にかけるという利用法では，たとえば，星団の運動をシミュレーションするということが議論されています．ある星団のn個の星すべてが及ぼしあう重力を計算して，星団がどのような動きをするか調べるわけです」

前回の蛋白工学研究所では10^{-9}秒から10^{-6}秒のオーダーで分子の運動をシミュレートしていたが，こちらは10^6年から10^9年のオーダーで星の運動を計算するという．スケールも，かたやオングストローム，かたや光年とまったく極端で，クラクラきてしまった．

星の生成過程はだいたい解明されたので，次は星団（銀河）が生まれた仕組みを探る段階．なんでも，星団の運動を計算する目的で並列コンピュータを開発しているグループもあるという話だ．

星団の次は，いよいよ宇宙生成のシミュレーション．と，考えたいところだが，まだそのプログラムを書くために必要となる基本的な方程式が揃っていない．物理学者は理論をよりエレガントなものにし，天文学者はいろんな現象や物体を探し解析することで，宇宙を作った法則に到達しようとしている．物理学の成果は天文学で応用され，天文学の成果で物理学が前進したり裏付けを得たりすることもあるだろう．

『ライフゲイムの宇宙』を読んで，法則（情報）が宇宙のすべてを生み出したと信じられるようになった．いつかはコンピュータ上に宇宙生成モデルを作れるに違いない．ただライフゲームの盤上ではムリそうだ．ノイマンの自己増殖機械を自然に生成させるには，10の後ろに10^{12}個ゼロが付くくらい，たくさんのピクセルが必要になるらしい．

特別付録 大型望遠鏡の正しい構え方

Astro Guide '90

コンピュータの中の小さな宇宙

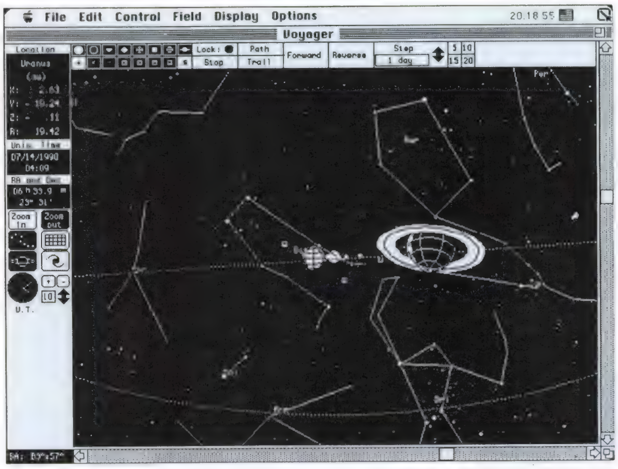
'88年は日米同時に高速天文シミュレーションが発表された年だ．７月にアスキーが「超高速天文シミュレーション」を発売し，遅れて年末にアメリカで発表になったのがCarina Software社の「Voyager」だ．天文ソフトはパソコンの初期の頃からさまざまなものが開発されてきたが，数値のみの出力であったり，貧弱なグラフィックの割には時間のかかるものばかりで，高速なグラフィック表示が可能なこの２つのソフトは画期的であったため，'88年は“高速天文シミュレーション元年”といえる．

本来シミュレーションというものは“if”の世界を実現するものだが，天文シミュレーションの場合少々事情が違う．どちらかといえば天文シミュレーションはデータベースの世界に近い．そこで求められるのは，“いつ”“どこで”“何が見えるか”だからである．ここで要求される条件は無数にあるので，計算で求めてしまうわけだ．そして計算結果を数値で表現するかグラフィックで表現するかで，その天文ソフトの性格が決定される．先の２つの天文ソフトは後者にあたり，精度をある程度犠牲にすることで高速化を実現している．とはいっても，パソコンのグラフィックに表示することを前提にすればたかだか４桁の精度があれば十分だ．そのおかげで，リアルタイム表示が可能となり，気軽に使えるソフトとして多くの天文ファンに受け入れられている．

さて，VoyagerはMacintosh用のソフトで，アメリカでは天文ソフトとして最高の評価を受けている．1,500個近くの恒星と星雲星団のデータを抱え，任意の時刻，観測地から見える星空はもちろんのこと，太陽系の軌道図や彗星などのデータ追加機能まで搭載し，およそ考えられる限りのシミュレーションが可能である．昨年Ver.1.2になり，惑星表面の経緯度表示や地上の風景エディタまで追加された．また，Voyagerの製作者たちは望遠鏡をコントロールできるスペシャルバージョンでカリフォルニアの砂漠でスターウォッチングを楽しんでいるそうだ．現在はまだ白黒表示だが，カラー化の噂も伝わっており，今後が楽しみなソフトだ．

今後両者ともさらにバージョンアップされていくだろうが，より多くの人にパソコンで天文を楽しんでもらうためには，現在の機能に加えて天文に関するさまざまな情報——星座の神話や星雲や惑星の写真など——を簡単に取り出せる機能が必要となってくるだろう．BGMやナレーション機能を付加するなどマルチメディアにはうってつけの題材であると思うのだが…….

(haru-k)



製品名：Voyager
価格：124.5ドル
問い合わせ先：
Carina Software
830 Williams Street San Leandro, California 94577

今月の星空

9月24日に水星が明け方の東の空で西方最大離角（17度52分）となり観望の好機となる．明け方の観測条件としては今年最高のものだ．

日の出のときの水星の高度は18度弱で，光度は0.5等級になる．

水星の約10度ほど下には−3.9等級の金星が輝いているので，これを目印にするとよい．

9月24日（月）　5時30分　東の空

また９月30日にはくじら座の長周期変光星ミラが極大になる．極大の光度は年によって違うが，昨年は２等級にまでなった．くじら座はなかなか見つけにくい星座だが，うお座のα星を目印にすると探しやすい．

第2回 EmacsとWnnを使う

プロフェッショナルワークベンチとしての UNIX入門

村井純

illustration/Yoshinori Otake

今回はテキストエディタについての話をします．テキストエディタというのは，ノートを取ったり，メモを書いたり，保存しておいた文書を後で直したりと，そういったことをワークステーション上で行なうためのツールです．

さて，テキストエディタの話をする前に，ワークステーション環境について知っておかなければならない予備知識がもう少しありますので，今回はそこから話を始めることにしましょう．

ファイルシステムとは?

ファイル

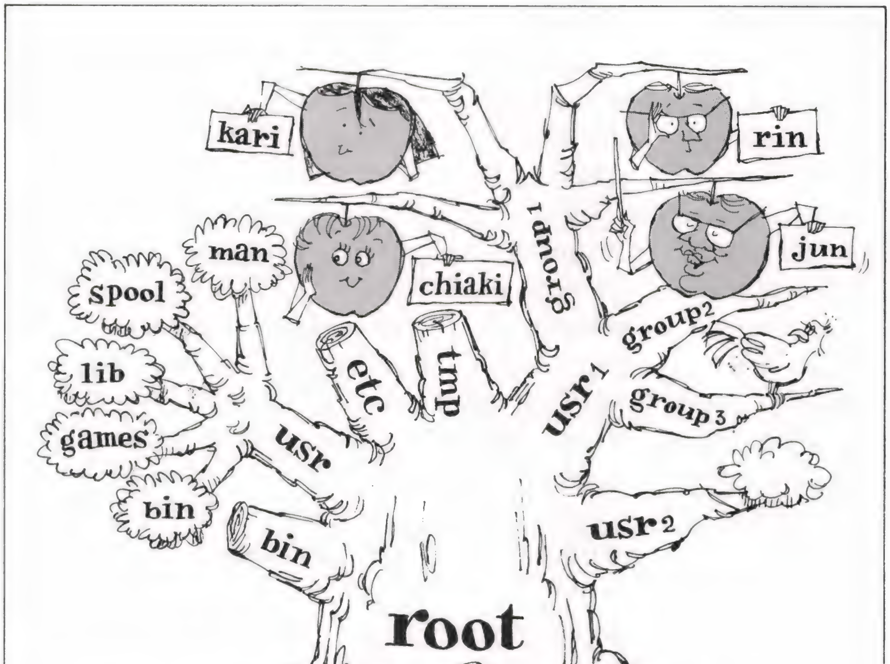
これからワークステーションを使って，レポートを作成したり，メールを受け取ったり，データを処理したりと，いろいろなことをしていくわけですが，それらの結果はワークステーション上（実際はファイルサーバマシンのハードディスク上）に保存されていくことになります．この保存される情報のかたまりを“ファイル”といい，ファイルの集まりを“ファイルシステム”といいます．

ワークステーションの中は，いくつもの部屋に分かれています．これらの部屋にもファイル同様1つ1つに名前が付けられています．ファイルに名前を付けようとするときは，そのファイルがある部屋の中にすでに存在している名前は付けられません．別の部屋に，同じ名前のファイルがあってもそれは構いません．

それでは，部屋の中にどんなファイルがあるかを見てみることにしましょう．ファイルのリストを見るにはls (list) というコマンドを使います(図1)．部屋の中は，さらに子部屋でいくつにも分割されていて，その子部屋の中にもファイルが入っているかもしれません．

ディレクトリ

これからワークステーションを使い続けていくと，授業のレポートのファイルや友人からのメールのファイルなどが作



られていくことでしょう．これらのファイルを整理せずに１つの部屋の中にどんどん入れていってしまうと，収拾が付かなくなってしまいます．つまり，玄関を入ると，そこに掃除道具から台所用品まで何もかもがあるような，いわば片付いていないワンルームマンションみたいな部屋になってしまうのです．

これでは，非常に使いづらいですから，部屋の中にさらにいくつかの子部屋を作り，そこにファイルを分類整理するということになります．この部屋のことを正式には“ディレクトリ”と呼びます．まず，今いるディレクトリ名を見てみましょう．

```
% pwd↲
/home/fs0210g/t90453hh
```

pwd (print working directory) は，今いる部屋の名前，つまりディレクトリ名を出力しなさい，というコマンドです．

今，pwdで出力されたディレクトリには，最後にログイン名が付いています．これが皆さんに与えられたディレクトリだということが分かるでしょう．この教室にあるどのマシンからログインしても皆さん用のディレクトリがありますね．これは，皆さんが使うディレクトリは，SFCのある場所にあってどのマシンからも同じものを見られるようになっているということです．

このディレクトリの構造は，ちょうど大きな団地があって，その中にマンションがあって，各戸の中には２DKや３DKというように部屋で仕切られている，というふうに階層的になっているのです．これが階層的なディレクトリ構造ということで，UNIXではこのような構造でファイルを管理しています（**図２**）．電話番号の場合も，市外局番-市内局番-加入者番号と階層的な構造を持っていて，階層を分かりやすくするために通常ハイフンを入れたりしますね．これと同じように，UNIXでは‘/’（スラッシュ）を階層の分け目として使います．

ディレクトリの歩き方

このように階層で作られる「木構造」が，UNIXファイルシステムの特徴です．ではいろいろなディレクトリを少し歩き

図１

```
% ls
Mail      Wnn       bitmaps   lectures  tmp
News      ascii     floppy    sources   wemacs
```

図２

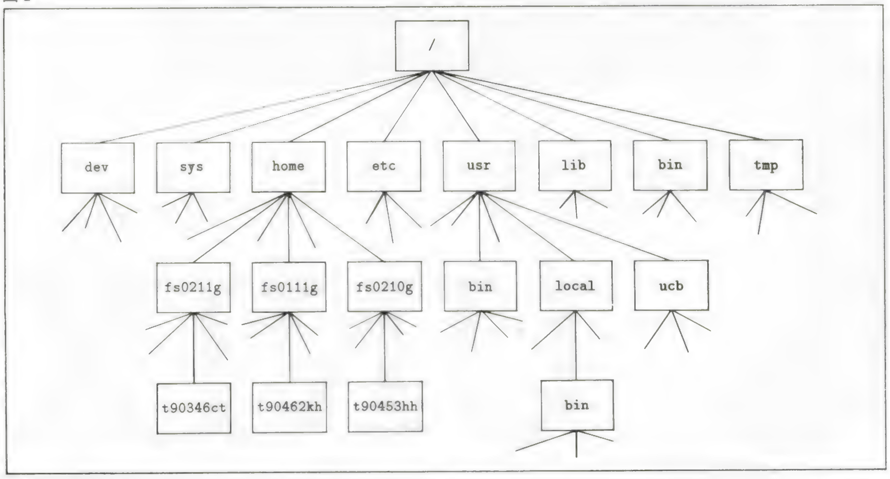


図３

```
% ls
lost+found  s90225as  s90429hh  t90045yi  t90273ms  t90501th
s90009ya    s90237hs  s90441yb  t90057mi  t90285as  t90513sm
s90021na    s90249ts  s90453th  t90069yi  t90297ts  t90525mm
s90033mi    s90261hs  s90465kf  t90081ku  t90309ts  t90537tm
```

回ってみましょう．ディレクトリを変えるには，cd (change directory) コマンドを使います．

```
% cd /home/fs0210g↲
```

ほかにどんなディレクトリがあるかlsコマンドで見てみると，ほかのユーザーのディレクトリ名が見えますね．その中には自分のディレクトリの名前もあります（**図３**）．

こうやってディレクトリを歩き回っていると，自分がどこにいるのか分からなくなり，迷子になってしまうことがあります．そんなときに，どこにいようと瞬時に自分のディレクトリに戻れる方法があります．cdの後に何も付けないで，リターンキーを押すのです．こうすれば，自分のディレクトリ (home directory) に戻れます．

それから，ディレクトリを変わるときに，/home/fs0210gとフルネームをいちいち打っていましたが，これは大変なので，いくつかの省略する方法があります．先頭を‘/’で始めずにディレクトリの名前を書けば今いるディレクトリ（pwdで出力されるディレクトリ）の１つ下のディレクトリを指定したことになります．たとえば，/homeにいるときに，/home/fs0210gとやれば，fs0210gというディレクトリにいくことになりますが，これは単に，

```
% cd fs0210g↲
```

でもいいのです．住所でいえば，“日本の”から始まって“東京の何区の何町”という言い方をするのが‘/’で始まる名前です．でも「原宿で会おう」というときには，もうすでに暗黙の前提の定義域“日本の東京にある原宿”があります．暗黙の前提の定義域があると，人間はすごく話しやすいですね．コンピュータにあいまいは困るので，必ず今いるディレクトリが暗黙の定義域となります．

また，今いるディレクトリは‘.’で表わすことができます．そして，すぐ上の階層のディレクトリを‘..’で表わします．ですから，/home/fs0210gから/homeに移りたいときには，

```
% cd ..↲
```

でもよいわけです．

また，先ほど自分のホームディレクトリに戻るのに単にcdと打てばよいと説明

しましたが,「自分のホームディレクトリ」は‘~’で表現することができます．また，ほかのユーザーのホームディレクトリは‘~ユーザー名’で示すことができます./home/fs0210g/t90453hhは，t90453hhさんにとっては‘~’であり，ほかの人にとっては‘~t90453hh’となるわけです．

自分のディレクトリの整理の仕方

さて，次に自分のディレクトリの中をどう整理するかを考えてみましょう．ホームディレクトリでlsコマンドを実行すると，WnnとかMailといったディレクトリが初めからありました．Mailというのは，次回講義する電子メールで送られてきた手紙をしまっておくためのディレクトリ，Wnnというのは，かな漢字変換の辞書とか，かな漢字変換に関する情報をしまっておくためのディレクトリです．これらは必ず使うものなので，あらかじめ作ってあるというわけです．

これからいろいろなことを練習するためにテスト用のファイルを作ります．それらは大概ゴミになるので，まずはそんなゴミファイルをしまっておくためのディレクトリを作っておきましょう．一時的なファイルをしまうディレクトリという意味でtmp (temporaryの略) を作ることにします．ディレクトリを作るためのコマンドはmkdir (make directory) です．

```
% mkdir tmp↵
```

逆にディレクトリを消すには，rmdir (remove directory) コマンドを使います．ただし，ディレクトリを消すときにはディレクトリの中にファイルやディレクトリが入っていてはいけません．ファイルを消すにはrm(remove)コマンドを使います．

```
% rm ファイル名↵
```

ファイルを消さずに別のディレクトリに移したいのなら，mv(move)コマンドを使います．

```
% mv ファイル名 ディレクトリ名↵
```

ディレクトリ名のところにファイル名を書けばファイル名を変更することになります．それに対して別のディレクトリにファイルをコピーしたいときは,cp(copy)コマンドを使います．

どういうふうにファイルを整理するかということはとても重要なことです．整理の仕方はまったくの自由ですが，どうやったら自分の使い方，自分の性格に一番合っているかを考えて，決めていくようにしましょう．

テキストエディタ Emacs

Emacsの基本的な操作法

では，これからファイルに文字を書き込んでいきましょう．そのための道具が始めに述べたテキストエディタです．できたファイルは先ほど作ったディレクトリtmpの下に入れることにしましょう．今回使うテキストエディタは“Emacs”というものです．ほかにも“vi”や“ed”といったテキストエディタがあります．

Emacsを起動するには，プロンプトから‘emacs↵’と打ちます．すると，Emacsがディスプレイをコントロールしている大家さんに「表示する領域を割り当ててください」と言ってきますから，マウスの左ボタンを押してEmacsのウィンドウを好きな位置に固定してください(**図4**)．

このEmacsのウィンドウ内は単なる紙だと思ってけっこうです．ここにキーボードから何か文字を打てばそれが入っていきます．リターンキーをたたけば次の行に移ります．

基本的な編集操作

文字を入力していくと,abcと書いたけれど実はacdと書き直したいとか，abcのaだけを消したい，といったような操作がどうしてもしたくなりますね．書いたり，消したり，戻ったり，1行だけ抜いたり，といったことが自由自在にできなくてはなりません．

カーソル移動

文字を入力するときはその文字の書いてあるキーを押せばよかったのですが，カーソルを動かすときはコントロールキーを使います．1行分上にいくのはCtrl-p (コントロールキーとpキーを同時に押す)，下にいくのがCtrl-n，1文字分右にいくのがCtrl-f,左にいくのがCtrl-bです(Previous, Next, Forward, Backward

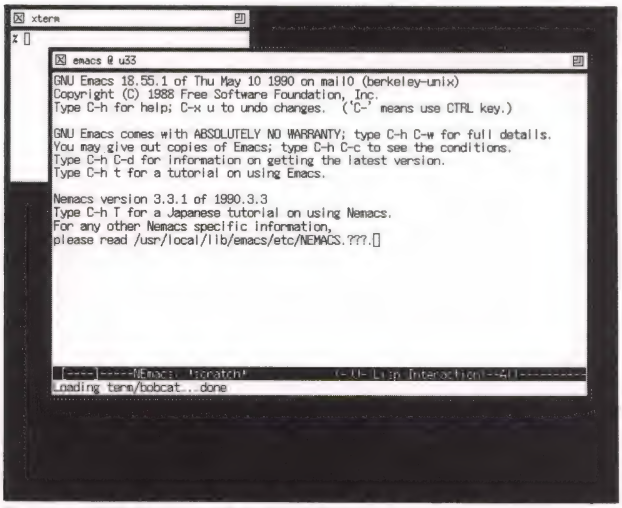


図4　Emacsを起動したところ.ディスプレイをコントロールしている大家さんに土地を割り当ててもらう.

図5　‘Ctrl-x 2’でウィンドゥを2つに分割する.分割したときは,同じ内容が両方のウィンドゥに表示されている.

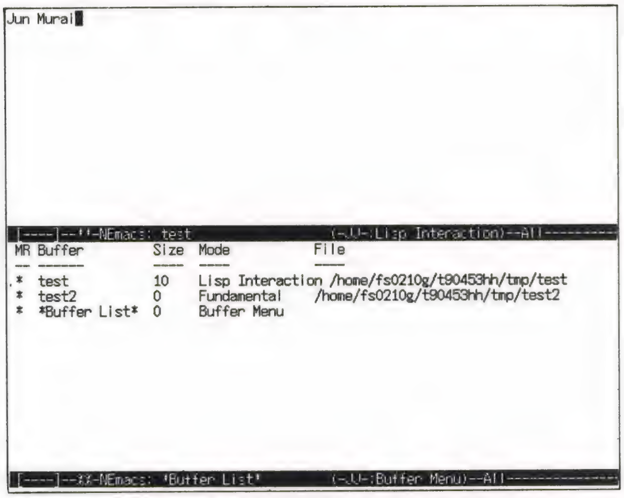


図６ 'Ctrl-x Ctrl-b'でバッファとその状態を見ることができる．今あるバッファは都合３個だ．

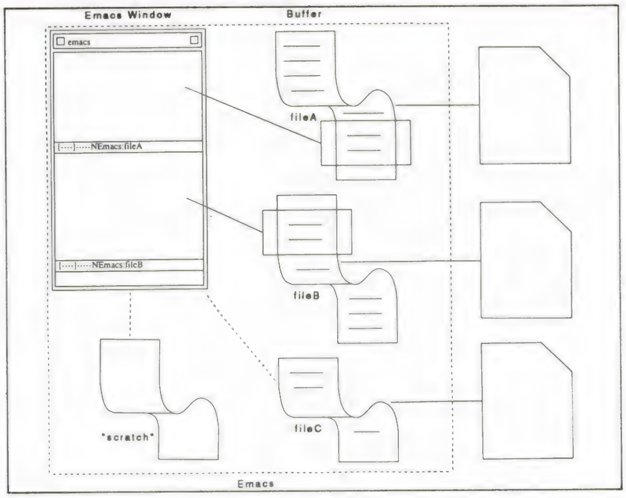


図７ ファイルとバッファとウィンドウの関係．バッファとファイルがまだ結び付いていないものもある．

と覚えてください)．また，Ctrl-aで行頭，Ctrl-eで行末に移動できます．

文字列中にカーソルがあるときにキー入力するとその位置に文字が挿入されていきます．たとえば，ab(b上にカーソルがある) とあるところでキー入力するとaとbの間に文字が挿入されます．

もう１つ覚えておかなければならないものに，メタキーがあります．EmacsではESCキーがこのメタキーに当たります．たとえば，Meta-xといった場合(メタエックスと呼ぶ)，ESCキーを押してから(押しながらではありません)xキーを押すことです．これもEmacsでよく使われる表現です．

で，ファイルの最後にいくのがMeta->です．そしてファイルの一番最初にいくのが，Meta-<です．カーソルの移動についてはとりあえずこのくらい覚えておけばよいでしょう．

削除

書いた文字を消したくなったとします．まず，カーソルの位置の１文字を消すのがCtrl-d (delete) です．カーソルのある位置からその行の最後まで消すというのが，Ctrl-k (Kill line) です．カーソルの前の１文字を消すのが，BackSpaceキーです(デフォルトの設定では，Ctrl-hはバックスペースではなくヘルプに割り当てられていますが，BackSpaceキーとCtrl-hが同じ働きをすることもあります)．

もっと広い範囲を一度に消したいときはどうするかというと，まず消したい範囲の一方の端(始めでも終わりでもよい)にカーソルを持っていってCtrl-Space Barでマーク（印）を付けます．その後カーソルを移動させてもう一方の端に持っていきます．このマークから現在カーソルのある地点までのことをリージョンといいます．リージョンはEmacsの大切な概念の１つですから覚えておいてください．リージョンを消去しなさい，というコマンドがCtrl-wです．

Undo!――間違えて消しちゃった! でも大丈夫

こうやってCtrl-wなどを使って消去していると，消さなくていいところまで間違えて消しちゃった，ということが必ず起こります．Emacsのスゴいのは，こういったときにやり直しができることです．Ctrl-_とやると直前の状態に戻れます．繰り返しCtrl-_をやっていくと，１回１回やったことの歴史をさかのぼっていきます．だから，あの時点に帰りたいと思ったら，そこまで何回もCtrl-_を実行すればよいのです．

また，Ctrl-kやCtrl-wで文字列を消した場合には，それらの文字列はすぐにはなくならないで，ある場所に取ってあります．ここから戻すときはCtrl-y (yank) とやります．だから，Ctrl-wでリージョンを消してから別のところにカーソルを持っていき，Ctrl-yとやるとそちらにリージョンを移動できるわけです．これを使って文字列のカット＆ペーストができます．

バッファとマルチウィンドウ

では，書いた文章を保存しておくにはどうするか，つまりファイルにするにはどうすればいいのでしょうか．

今まで行なってきた作業は，Emacsという道具があって，これが用意してくれた画面にいろいろな文字を書き込んでいくということでした．これをバッファといいます．Emacsで作業をするときに使用する紙がバッファだと考えてください．作業が終わったら必要に応じてその内容をファイルに書き出したりするわけです．

今書いた文章をホームディレクトリの下のtmpというディレクトリの下に，testという名前のファイルで保存するということは，バッファの中身を~/tmp/testというファイルに書き出すということにほかなりません．

Emacsはウィンドウを開いて，どのバッファのどの部分をディスプレイに出す

図 8　Wnnを使った変換の例

```
|わたしのなまえはむらいじゅんです|
[ aあ]--**-NEmacs: *scratch*          (JJJ-:Lisp Interaction)--All---------
    ⇩  スペースバーを押して漢字に変換する.
|私の 名前は 村井 淳です|
[漢字]--**-NEmacs: *scratch*          (JJJ-:Lisp Interaction)--All---------
    ⇩  Ctrl-fでカーソルを１文節ずつ右に移動する.
|私の 名前は 村井 順です|
[漢字]--**-NEmacs: *scratch*          (JJJ-:Lisp Interaction)--All---------
    ⇩  Ctrl-kでカーソル位置の前までを確定させる.

私の名前は村井|じゅんです|
[ aあ]--**-NEmacs: *scratch*          (JJJ-:Lisp Interaction)--All---------
    ⇩  Meta-sで漢字を一覧表示させる.
私の名前は村井|淳です|
[漢字]--**-NEmacs: *scratch*          (JJJ-:Lisp Interaction)--All---------
次候補:  0.順です 1.淳です 2.準です 3.純です 4.准です 5.潤です 6.純です
    ⇩  正しい漢字を指定して，リターンキーで確定させる.
私の名前は村井純です
[ aあ]--**-NEmacs: *scratch*          (JJJ-:Lisp Interaction)--All---------
```

かということをコントロールしています．つまり，バッファはちょうど長い巻き物のようなものだと思えばよいでしょう．Emacsのウィンドウはその一部を見せてくれているわけです．Emacsはウィンドウを分割して，それぞれに別のバッファを表示することも可能です．バッファは一度の作業でいくつも使用できます．

バッファとウィンドウの操作

バッファ全部をファイルに書き出すには，Ctrl-x Ctrl-wとやります．すると，エコー領域にWrite file:と出て，ファイル名をどうするか聞いてきますから，ファイル名を指定します．たとえば，Emacsをディレクトリ~/tmpで立ち上げると，Write　file:~/tmp/と表示されますから，あとはtest↵と入力すれば，tmpの下にtestというファイルができるわけです．

現在Emacsのウィンドウは，１つのバッファを見せるために使われています．それでは，今度は２つのバッファを表示してみましょう．Ctrl-x 2とやると，ウィンドウが２つに横分割されます（**図５**）．Ctrl-x 2を何回もやるとウィンドウがどんどん分割されていきます．ウィンドウが分割されているときに，文字がどのウィンドウに入力されるかはカーソルがどのウィンドウにあるかによって決まります．ウィンドウ間でカーソルを移動させるのはCtrl-x oです．

この１つ１つのウィンドウは好きなようにバッファと結び付けることができます．あるウィンドウにいって，Ctrl-x Ctrl-fとやると，Find file:といってファイル名を聞いてきます．つまり，あるウィンドウに結び付けられているバッファにファイルを読み込もうとしているのです．ここでは，test2と指定してみましょう．test2というファイルはまだ存在していませんから，(new file)といって実際にはないファイルの内容をウィンドウに表示してくれます．何も入っていないファイルですから，ウィンドウ内は真っ白です．これでtestというファイルが一方のウィンドウに表示され，もう一方のウィンドウにはtest2という空のファイルが表示されるようになります．

もうtest2は見なくていいからtestだけを表示するようにしたいと思ったら，testの表示されているウィンドウにカーソルを持っていってから，Ctrl-x 1とやるとtestのウィンドウだけを表示するようになります．test2はただ画面に表示されていないだけで，バッファが消えてしまったわけではありません．ですからもう一度ウィンドウを分割して，片方のウィンドウにこのバッファを見せてといえばまたtest2を見ることができます．

ほかにどんなバッファがあるのか見たいときはCtrl-x Ctrl-bと打ちます．すると，今あるバッファの名前とその状態のリストを見ることができます（**図６**）．これで見ると，バッファは３つあって，test2は何も書いていないので，サイズはゼロとなっていますね．最後の項目はそのバッファがどのファイルと結び付けられているかを示しています．Emacsを立ち上げたばかりでバッファとファイルが結び付けられていないときは，バッファの名前は*scratch*になっています．*Buffer List*というバッファは，このバッファリストを表示しているバッファ自身のことです．バッファをファイルと結び付けなさい，というコマンドがCtrl-x Ctrl-wです．これに対して，Ctrl-x iとやると，ファイルとバッファの結び付きはそのままで，別のファイルの中身をこのバッファに挿入します．たとえば，test2が表示されているウィンドウにカーソルがある状態でCtrl-x iとやるとInsert file:~/tmp/と聞いてきますからそこで，testと指定するとこのファイルの中身をバッファtest2に取り込むことができます．

Emacsのファイルとバッファとウィンドウとの関係というのは，複雑な仕組みではあるけれども，大変便利な機能です（**図７**）．Emacs上でいろいろなプログラムを動かして，バッファにこのプログラムの実行結果を書き出させるといったこともできます．この機能を使えば，電子メールや電子ニュースをEmacs上で読み書きしたりもできるわけです．

日本語入力プログラム Wnn

漢字変換

これでEmacsで文字列の操作ができるようになったと思います．さて，もう１つ文章を作るうえで大切なことに「日本語を入力する」ということが挙げられます．日本語入力には，“Wnn”という漢字入力プログラムを使います．また，Emacs上でWnnを使うためのシステムを“Egg”と呼びます．

Ctrl-＼でWnnが立ち上がり，日本語入力モードになります．キーボードによっては，‘＼’がないかもしれませんが，その場合は‘¥’でも同じです．今まで［----］だったEmacsのウィンドウの一番下の黒い棒（モード行）の左端が［ aあ］に変

わります．これが，「日本語入力モードになったよ」という合図です．

この状態でローマ字入力していくと自動的にひらがなになっていきます．たとえば，“murai”と打つと“｜むらい｜”というふうに表示されるはずです．これを漢字に変換するにはスペースバーをたたきます．すると，モード行が［漢字］になって，“｜むらい｜”が“｜村 い｜”となりました．まず，両側が縦棒で囲ってあるときはまだ漢字が確定していないということです．この“村”と“い”の間の空白に注目してください．このスペースで区切られた単位で辞書を引いています，ということなのです．つまり，“むら”と“い”で辞書を引きにいった，ということが分かるのです．この空白で区切られたひと固まりを文節といいます．この場合は，“むらい”で1つの文節として辞書を引いたほうがいいわけです．そこで，“むら”と“い”を別々にせずに，“むらい”で辞書を引いてください，つまり文節を1文字分伸ばしてくださいとWnnに命令するには，Ctrl-oとやります．逆に文節を縮めるのはCtrl-iです．こうやって漢字に変換していきます．Ctrl-mまたはリターンキーをたたくと両側の縦棒がなくなって文字列が確定します．

入力間違いをしてしまった場合はどうすればよいでしょうか．たとえば，“つかえかな”と入力したかったのに，“つかへるかな”と入力して変換し，“｜津 化 減るか 名｜”となってしまったとします．こういったときには，Ctrl-gを押すと，また“｜つかへるかな｜”に戻りますから，間違いを訂正してもう一度変換し直せばいいわけです．

また，たとえば“｜わたしのなまえはむらいじゅんです｜”が“｜私の 名前は 村井 淳 です｜”と変換されたとしましょう．この場合，Ctrl-fを押すことによってカーソルを1文節ずつ右に進ませることができます．逆にCtrl-bで1文節ずつ左に戻せます．変換中の文の最初の文節に戻すにはCtrl-a，最後の文節の頭まで進めるのはCtrl-eを使います．Emacsのカーソル移動と同じ要領ですね．“じゅん”が間違っていますから，ctrl-fでカーソルを進めて，スペースバーかCtrl-nを何度か押せば“じゅん”の候補が次々にでてきます．Ctrl-pで前の候補に戻ることもできます．“じゅん”と読む漢字の候補がたくさんあって，何度もスペースバーを押すのが面倒だと思うならばMeta-sです．すると，エコー領域に，“じゅん”と読む漢字の一覧がでてきますから，数字を指定するか，カーソルを移動させてリターンキーを押すことによって漢字を指定します．縦棒で囲まれている文を全部確定するときにはリターンキーまたはCtrl-mを使いましたが，カーソルより前の文節だけを確定する場合はCtrl-kを使います（図8）．

Wnnの辞書

うまく漢字に変換できるかどうかは，大部分が辞書にかかっています．辞書が上手に作られていると，上手な変換ができます．Wnnの場合，どうやって辞書を作ったかというと，まずWnnを日本全国にタダでばらまいて使ってもらったのです．その人たちは自分がよく使う単語を辞書に登録していきますね．ある時点で，ネットワークを使ってその辞書を集めます．それらを統計処理をして，単語を使われている順に並べて新しい辞書を作るというわけです．そして，またこの新しい辞書が配布されるのです．だから，たとえば“千秋”という名前がまだ辞書に登録されていなかったとしたら，ここで登録しておけば，この次に配られる辞書には“千秋”という名前が登録されていることになります．

辞書に単語を登録するには，登録したい部分をリージョンとしてから，Meta-x toroku-regionとしてください．すると，読みの入力を要求してきますから，これに従って読みを入力し，続いて品詞を設定してやると，自分の“プライベート辞書”にその単語が登録されるようになるわけです．

表1 Emacsの主なコマンド

カーソル移動	Ctrl-p…1行前に Ctrl-n…1行後に Ctrl-b…1文字前に Ctrl-f…1文字後に Ctrl-a…行頭へ Ctrl-e…行末へ Meta-<…ファイルの先頭へ Meta->…ファイルの最後へ
消去	Ctrl-d…カーソルの下の文字を消す BackSpaceキー…カーソルの前の文字を消す Ctrl-k…カーソルからその行の最後まで消す Ctrl-w…リージョンを消す
リージョンの設定	Ctrl-SpaceBarで付けたマークからカーソルのある位置まで
ファイル操作	Ctrl-x Ctrl-w…バッファの内容をファイルに書き出す Ctrl-x Ctrl-f…ファイルの読み込み Ctrl-x Ctrl-s…ディスクへ書き込む
マルチウィンドウ	Ctrl-x 2…ウィンドウを分割する Ctrl-x o…ウィンドウ間のカーソル移動 Ctrl-x 1…ウィンドウを1つにする Ctrl-x b…表示するバッファの切り替え Ctrl-x Ctrl-b…バッファのリストを表示 Ctrl-x i…バッファにファイルを挿入する
その他	Ctrl-_…Undo（やり直し） Ctrl-y…Ctrl-kやCtrl-wで削除したテキストの取り込み

表2 Wnnの主なコマンド

Ctrl-¥ …日本語入力のON，OFF	Ctrl-g …やり直し
SpaceBar …変換	Ctrl-mまたはリターンキー …確定
Ctrl-o …文節を伸ばす	Ctrl-k …カーソルより前を確定
Ctrl-i …文節を縮める	Meta-s …変換候補の一括呼び出し

OS/2 アーキテクチャ入門

OSは，CPUやメモリ，ディスクなど，さまざまなシステムの資源を管理し，その上で動くアプリケーションプログラムに対して，それらを操作するためのサービスを提供する．当然のことながら，この“OSが提供するサービス”の種類や形態は，アプリケーションの構造に大きく影響を及ぼすこととなる．

そこで今回は，アプリケーションプログラムの実行環境として，OS/2がいったい何を提供するのか，またデバイスの管理など，OS/2の入出力機構がどのようになっていて，アプリケーションプログラムがどのような恩恵を受けられるのか，といった点を解説しようと思う．

第2回

プログラムの実行環境

佐藤和彦
山田伸一郎

プログラムの形態

OS/2のプロテクトモードで動作するプログラムは，“カーネルアプリケーション”と“PMアプリケーション”に大きく分類される．これらは，ユーザーインターフェイスなど，外見上の違いもさることながら，プログラムの内部構造も異なったものになる．この違いは，それぞれの実行環境の違いに由来する．異なる環境から異なる文化が生まれるように，環境の違いはその中で活動するものに影響を及ぼす．まず始めは，実行環境の違いという視点に立ち，カーネルアプリケーションとPMアプリケーションの違いを概観してみよう．

カーネルアプリケーション

カーネルアプリケーションは，OS/2カーネルが提供するサービス(カーネルAPI)だけを利用して動作するプログラムである．カーネルAPIの一覧を**表1**に示した．表からも分かるように，OS/2のカーネルAPIは，int 21Hなどのソフトウェア割り込みで提供されていたDOSのシステムコールに相当するものだ．もちろん，プロセス管理やメモリ管理など，DOSにはなかった機能が大幅に強化されている．ただし，カーネルAPIにはグラフィック関連の機能は含まれていないので，カーネルアプリケーションのユーザーインターフェイスは，テキストベースになる．

OS/2のカーネルAPIは，Cの標準関数の機能を包含しているので，すでに作成されたCプログラムを移植するのは容易だ．移植作業の大半は，OS/2用に提供されたライブラリをリンクし直すことによって解決する．単にライブラリの再リンクでは解決しない場合も，適当なカーネルAPIを呼び出すように変更するだけでよい．

また，カーネルアプリケーションのプログラミングスタイルは，従来のCプログラムと非常によく似ている．処理の基本的な流れは，ユーザーからの入力が必要になった段階でカーネルAPI経由で入力を取得し，処理を行ない，結果を(画面などに)出力するものだ．OS/2では，マルチスレッドなどの新しい概念が導入されたが，たとえ複数のスレッドを利用するプログラムであっても，この基本的な処理の流れは変わらない．入出力の競合が

表1 カーネルAPIの一覧

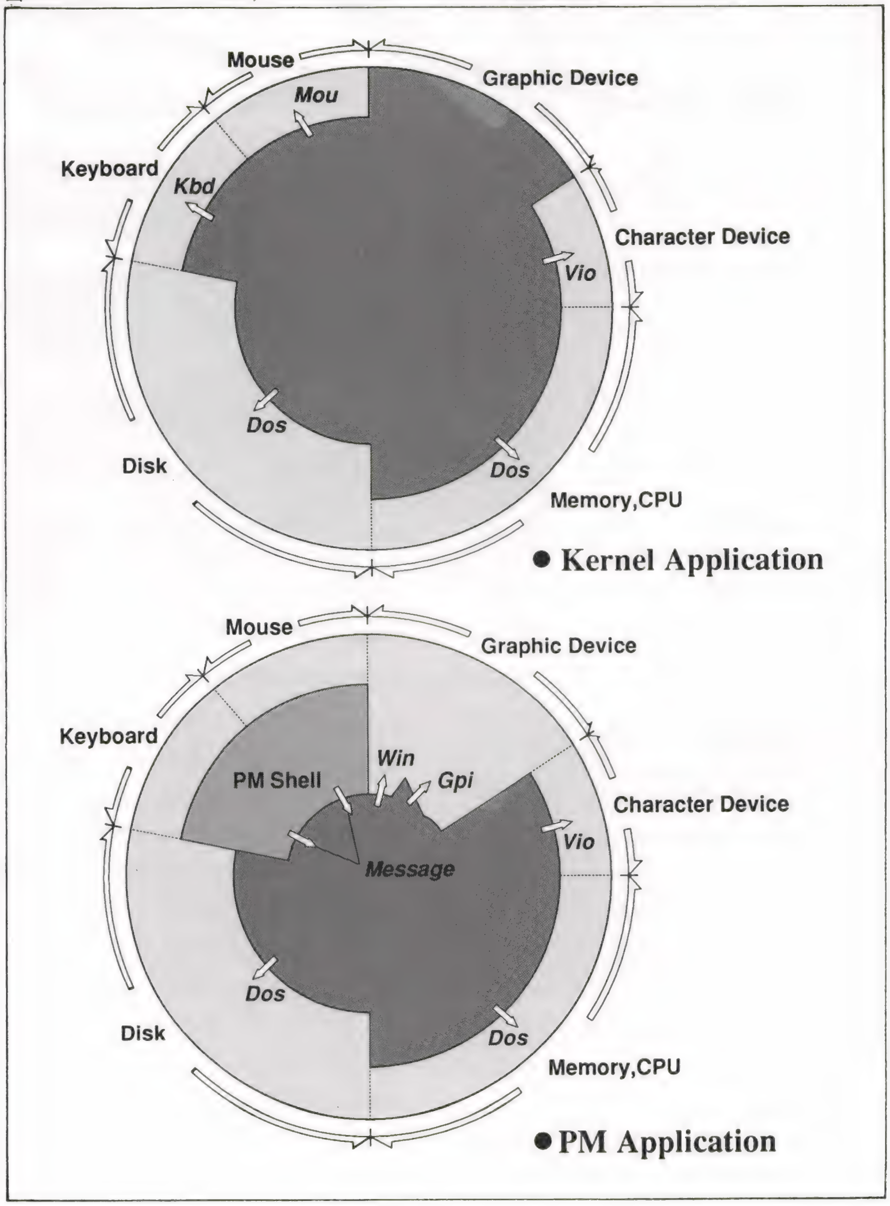
	内容	PM環境での使用
Dos	ファイル入出力，メモリ，プログラム管理など	○*
Vio	キャラクタディスプレイ入出力	一部使用可
Kbd	キーボード入出力	×
Mou	マウス入出力	×

*PM環境では，キーボード，マウスをオープンして入出力を行なうことはできない．

表2 ファミリAPIの抜粋

API名	機能	対応するDOSシステムコール（int21H）
DosOpen	ファイルのオープン	ファンクション3DH
DosRead	ファイル読み出し	ファンクション3FH
DosExit	プロセスの終了	ファンクション4CH

図1 カーネルアプリケーション，PMアプリケーションとシステム資源の関係



起こらないように，セマフォを使った排他制御などの配慮が必要になるだけだ．

●DOSボックスとバイモーダルプログラム

OS/2には，従来のDOSアプリケーションをそのまま実行できる環境として，DOSボックスと呼ばれるものが提供されている．DOSボックスでは，DOSのシステムコールだけを使用したアプリケーションの完全な動作が保証されるばかりでなく，グラフィック操作など，ハードウェアを直接制御しているようなDOSアプリケーションもそのまま実行できる場合が多い．

また，OS/2のカーネルAPIのうち，DOSのシステムコールでエミュレート可能なものをファミリAPIと呼ぶ（**表2**）．このファミリAPIのみを使用したアプリケーションプログラムを，BINDというユーティリティで加工すれば，DOSまたはDOSボックス（リアルモード）とOS/2（プロテクトモード）のどちらの環境でも動作する.EXEファイルを作成できる．このように，どちらのモードでも実行できるプログラムは，バイモーダルプログラム（Bimodal Program）と呼ばれる．いうまでもなく，バイモーダルプログラムがOS/2上で実行されるときは，カーネルアプリケーションになる．

PMアプリケーション

従来のスタイルを踏襲したカーネルアプリケーションに対し，まったく異なったプログラミングモデルに従うのがPMアプリケーションである．カーネルアプリケーションでは，カーネルAPIを呼び出すことによって，明示的にユーザー入力を取得していた．しかしPMアプリケーションでは，ユーザーからの入力が“メッセージ”という抽象化された形でプログラムに通知される．PMアプリケーションは，メッセージ駆動型（イベント駆動型）と呼ばれるプログラミングモデルに従うことになる．

●PMアプリケーションのプログラミングモデル

今述べたように，メッセージ駆動型のプログラミングモデルでは，プログラムは明示的には入力の取得を行なわない．正確には，“行なってはならない”のである．これはなぜだろうか．

ご存じのように，PMはマルチウィンドウ環境である．PM環境では，複数のウィンドウを開くことによって，複数のプログラムを，同時に実行することができる．しかし，たとえばキーボードやマウスなどの入力デバイスは，システムに1つしかない．つまり，ある時点でそれらの入力デバイスを所有できるウィンドウは，ただ1つしかない．今，ユーザーによってキー入力が行なわれたとしよう．この入力情報は，いったいだれが（どのウィンドウが）受け取ったらよいのだろうか．もちろんユーザーは，現在どのウィンドウがアクティブ（入力を受け付ける状態）なのかを知っている．しかしプログラムは，現在自分がアクティブなのか，そうでないのかということは分からない．

この問題の解決策の1つは，あるプログラムがマウスやキーボードの入力を一手に引き受け，その後，適当なアプリケーションにその入力情報を通知するという方法を採ることだ．ここでいう“ある1つのプログラム”は，PM環境ではPMシェルと呼ばれる．PMアプリケーションは，キーボードやマウスを資源として所有する必要はない．これら入力デバイスの管理は，PMシェルが一括して行なってくれる．

図1に，カーネルアプリケーション，PMアプリケーションと，さまざまなシステム資源の関係を示した．この図では，円の外周に個々のシステム資源が配置されている．そして円の中心に向かうほど，アプリケーションが受けられるサービスが高機能になる．

●メッセージキュー

マウスやキーボード入力は，PMシェルからPMアプリケーションに送られる．この通信は，メッセージキューという待ち行列を通じて行なわれる．キーの押下／開放，マウスの移動，そしてマウスボタンの押下／開放などのユーザーからの入力の断片に応じて，キューの要素（メッセージ）が作成される．そしてアプリケーションプログラムは，このキューを見張り，キューから1つの要素を読み出しては，対応する処理を行なう．

メッセージキューに格納されるメッセージは，入力に関するものだけではない．ウィンドウの描画やタイマなど，さまざまなイベントがメッセージとして，キューを通じてアプリケーションプログラムに送られる．

プロセスの起動から終了まで

OS/2下のプロセス（プログラム）が起動されてから終了するまでには，どのようなことが起こるのだろうか．次は，プロセスが起動されてから終了するまでの間に，OS/2がプロセスに対して提供するさまざまなサービスを，順を追って紹介してみよう．

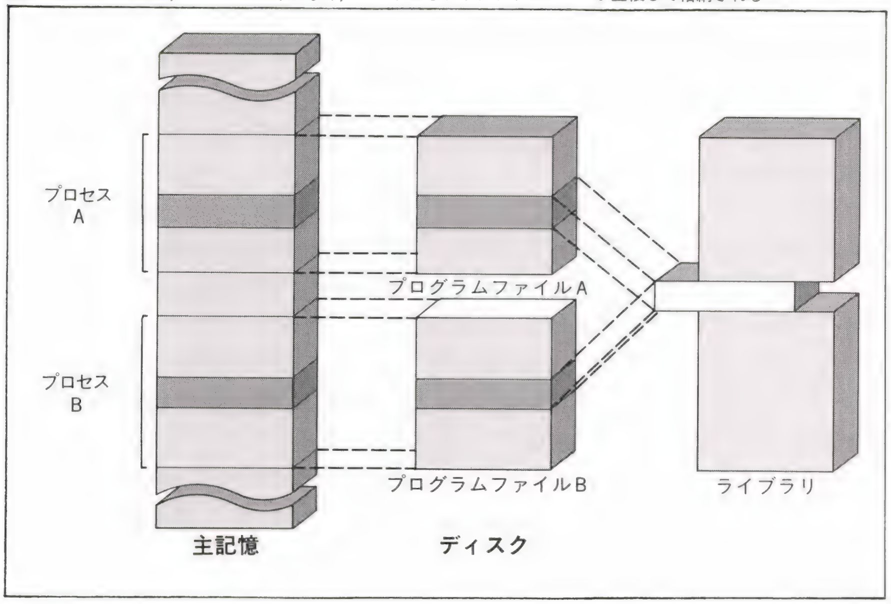
これまで，アプリケーションの実行形態としては2種類があると述べた．しかし，カーネルアプリケーション，PMアプリケーションともに，OS/2下の“プロセス”であることに変わりはない．そこで，まず初めはプロセス一般の起動から終了までに関して述べることにする．そして最後に，プロセスがPMアプリケーションに変わるタイミングについて触れよう．

プロセスの誕生

アプリケーションプログラムが起動されると，OS/2はそのプログラムの格納されたファイルをディスクからメモリに読み込む．このファイルは実行形式ファイル（.EXEファイル：実行可能ファイル）と呼ばれる．実行形式ファイルには，プログラムのコードやデータだけでなく，プログラムを動作させるために必要なメモリ容量などの情報も記録されている．

メモリに読み込まれたプログラムは，プロセスとして実際に実行を開始する以前に，それを起動したプログラム（通常はCMD.EXE）からファイルハンドルや環境変数などの情報を引き継ぐ．その後，スレッドが割り当てられ（OS/2の実行単位はスレッドであった），プロセスが誕生するのである．

図2　静的リンクでは，ディスクだけでなく，メモリにもライブラリのコピーが重複して格納される



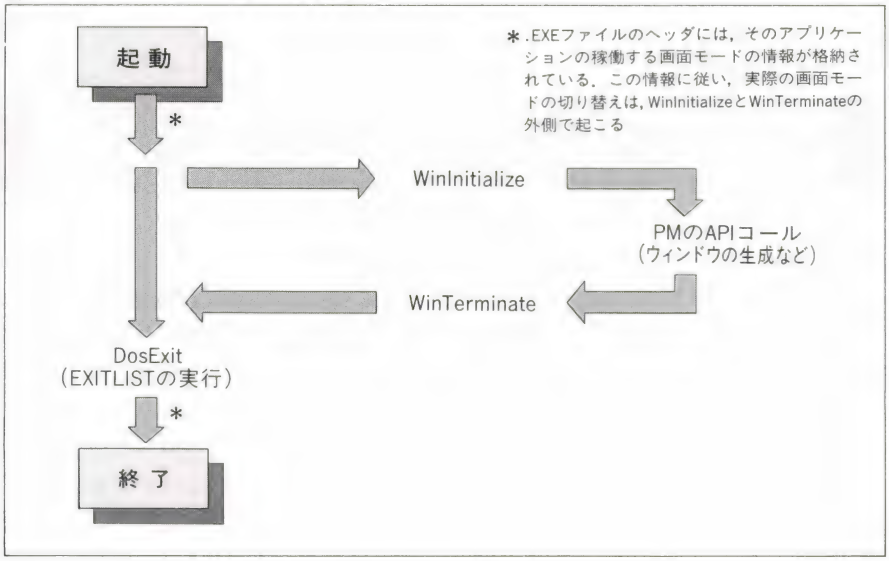
●プログラムファイルとコード

通常，プログラムの実行中は，プログラムが参照するすべてのコードはメモリ上にあると考えるのが一般的だ．そのため，プログラムに必要なコードはすべてプログラムファイルの中に組み込み，各モジュールの参照関係を解決しておくという手段が広く用いられる（システムコールのような特殊な“参照”やオーバーレイのような例外もある）．この場合，参照関係の解決（リンク）はプログラムファイルの生成時に処理され，プログラムを実行する前に固定されるので，“静的リンク”と呼ばれる．

しかし静的リンクでは，複数のプログラムファイルが同じライブラリルーチンのコピーを含んでいるという無駄が生じる．比較的大容量のディスクデバイスでは，この程度の無駄には目をつぶることもできる．しかし複数のプログラムが実行されるマルチタスクシステムでは，この問題はメモリの無駄遣いとなり，同時

OS/2
アーキテクチャ入門

図3 PMアプリケーションの起動から終了まで



に実行できるプログラムの数の制限に直結する（**図2**）．

●動的リンク

“動的リンク”（Dynamic Link）は，参照関係の解決をプログラムのロード時以降に行なう手法で，OS/2のAPI関数のほとんどは，この動的リンクによってアプリケーションと結合される．

動的リンクでは，プログラムに必要なすべてのコードを，そのプログラムファイル中に組み込んでおく必要はない．システムがプログラムをロードするときに，必要なコードがファイル中になければ，まず，そのコードがすでにメモリ上にあるかどうかが調べられる．もしあれば，プログラムの参照は，そのコードによって解決される．メモリ上にもなければ，ライブラリファイルを検索し，必要なコードをメモリにロードする．このライブラリは，ダイナミックリンク・ライブラリ（DLL：Dynamic Link Library）と呼ばれる．

この動的リンクは“ロード時リンク”と呼ばれるものである．これに対しプログラムが必要なコードを，必要になった時点で自分でロードし，不要になれば破棄する“実行時リンク”も可能だ．さらに，ロード時リンクをそのコードが必要になるまで遅らせて“実行時リンク”に近い機能を自動化することもできる．

●プライマリスレッド

プログラムがロード，動的リンク，環境の引き継ぎが完了すると，いよいよプログラムの実行が始まる．つまり，そのプログラムを実行するために1つのスレッドが割り当てられ，プロセスがスタートするわけだ．このスレッドはプライマリスレッドと呼ばれる．プライマリスレッドは別のスレッドを生成し，自分のプロセス内のコードを，複数のスレッドにより非同期に実行させることができる．

プロセス内のすべてのスレッドは処理が終わった時点で個々に終了できるが，プライマリスレッドの終了だけは特別な扱いになっている．プライマリスレッドの終了はプロセス全体の（すべてのスレッドの）終了を意味し，逆にいえばプロセスが生きている限り，プライマリスレッドは存在し続けるのである．

●終了時の処理とEXITリスト

プロセスは，その実行を終了する以前に，自分がシステムから得たメモリブロックやセマフォなどのシステムの資源を解放し，システムに返さなければならない．たとえば，オープンしたファイルをクローズするなどである．これらの資源の後始末を行なわずにプロセスが終了してしまうと，システムの資源が枯渇したり，システム全体の整合性が崩れてしまったりする．もちろん，こうした資源の後始末はそれを使用したプロセス自身で行なうのが理想的だ．しかしOS/2では，資源の後始末が確実に行なわれるように，EXITリストと呼ばれる機構が用意されている．EXITリストは，手続きのアドレス（特殊な関数のエントリポイント）のリストで，プロセスが終了するとき（DosExitをコールしたときなど）に，それらの手続きが順番に実行される．システムは，プロセスがなんらかのシステム資源を要求したとき，これを解放するルーチンをEXITリストに登録しておく．また，アプリケーション自身がこのEXITリストに“後始末ルーチン”を登録することも可能だ．たとえばテンポラリファイルの削除を，EXITリストを利用して行なうようなことが可能となる．

プロセスが終了するときには，EXITリストは必ず実行されなければならない．先に述べたように，プロセスが生きている限り，必ずプライマリスレッドが存在するようになっているのは，プライマリスレッドによってEXITリストが実行されるようにするためである．

●強制終了

プロセスは常に自分の処理が完了したときにだけ終了するのではなく，外部からの操作によって終了させられることもある（ユーザーがCTRL-Cを入力した場合など）．このときプロセスは，システムによって強制的に破棄されてしまうのではなく，“ただちに終了せよ”という意味のシグナル（終了シグナル）を受け取る．そして，デフォルトのシグナルハンドラによって，プロセスは自分自身でその実行を終了する．つまり，プロセスは“殺す”ものではなく，“自殺してもらう”ものなのだ（このようにプロセスを自分自身で終了させることの最大のメリットは，EXITリストを実行できるということにある）．

また，プロセスを外部から強制的に終了させる手段である終了シグナルの効果を保証するためには，そのシグナルがプロセスに確実に通知されるようにしておかなければならない．OS/2では，プロセスにシグナルが送られると，プライマリスレッドに制御が移り，プライマリスレ

ッドがシグナルハンドラを実行するようになっている（前述したように，プロセスが生きている限り，プライマリスレッドは生存している）．

●PMアプリケーションの誕生

プロセスの起動時は，カーネルアプリケーションとPMアプリケーションの区別はない．親プロセスからの環境の引き継ぎなどは，まったく同様にして行なわれる．それでは，起動されたプロセスが，PMアプリケーションとなるのはいつなのだろうか．

それは，プログラムが“WinInitialize”というAPIをコールした瞬間である．このときから，プログラムはPMアプリケーションとして認識され，GpiやWinなどのグラフィックAPIをコールできるようになる．そして，PMアプリケーションは，“WinTerminate”によって処理を終了する．WinTerminateの役割は，WinInitializeやWin／GpiなどのAPIによって割り当てられたメモリや，メッセージキューなどを解放することである．

PMアプリケーションのプロセスが終了するときにも，やはりEXITリストが実行される．PMアプリケーションがWinInitializeをコールすると，このEXITリストにWinTerminateに相当するものが登録される．したがって，なんらかの理由でWinTerminateをコールせずにアプリケーションが終了しても，必要な後処理は確実に行なわれるようになっている（**図3**）．これは便利な機能だが，原則としては，アプリケーションが取得した資源は，そのアプリケーション自身の責任で解放しなければならない．

資源の管理と抽象化

プロセスとして生命を得たアプリケーションは，その生涯を終えるまでの間に，さまざまなデバイスと入出力を行なうことになる．しかし，メモリやディスクなど，プロセスから利用できる資源には限りがある．したがってOSには，システムを効率よく運用していくために，こうした資源を適切に管理する機構が必要になる．また同時に，さまざまなデバイスの特性の違いを吸収し，操作しやすい形でアプリケーションに入出力機能を提供することも期待される．

次は，アプリケーションプログラムがさまざまなデバイスを効率よく扱えるように，OS/2がどのような機構を備えているのか，またそれによってアプリケーションが享受できる恩恵は何か，といったことを述べよう．

ディスクデバイス

物理的なディスクデバイスの操作は，適当なパラメータを設定して，ディスクに対してI/O命令を送出することである．この“パラメータ”は，ハードウェアごとに異なることも多い．もちろん，プログラムからこのI/O命令を送ってディスクを操作することは可能だが，通常，上位のプログラムからディスクをアクセスする場合には，ファイルの名称など，より抽象的な表現を利用するのは周知のとおりである．そしてプログラムが行なった“ファイル入出力”は，最終的にはOSによってディスクのI/O命令に変換される．この“変換”を行なうソフトウェアシステムが，ファイルシステムである．

ファイルシステムは，OSの重要な構成要素の1つである．特に，OS/2のようにメモリのスワップや動的リンクなどでディスクアクセスを頻繁に行なうOSでは，ファイルシステムの性能が，システムの総合的なパフォーマンスを大きく左右する．

●HPFSと拡張属性

最近では，パーソナルコンピュータにもハードディスクが普及し，さらにその大容量化が進んでいる．ご存じのように，DOS（および，OS/2 Ver.1.1まで）が提供するファイルシステムはFAT（File Allocation Table）ファイルシステムと呼ばれる．このFATファイルシステムは，FATと呼ばれるテーブルに，ディスク内の全クラスタに1対1に対応するエントリがあり，このテーブルをもとに領域の確保や解放，ファイルアクセスなどを行なうものだ．小規模なファイルシステムでは，このFATファイルシステムで高速にディスクアクセスを行なうことが可能である．しかしFATファイルシステムは，そもそも，フロッピーディスクを効率よく使用するために設計されたファイルシステムである．いまや数百メガというオーダーに達したディスクをFATファイルシステムで使用するのは，容量の面でも，アクセス速度の面でも，限界に至っているといってよいだろう．この問題を解決するためにOS/2 Ver.1.2では，FATファイルシステムに加え，新しいファイルシステムとしてHPFS（High Performance File System）が提供された．

HPFSでは，ファイルの割り当て単位が512bytesなので，小さなファイル（1Kbytes前後）が数多く存在しても利用効率が極端に悪化したりはしない．

またファイル名や物理的な割り当て位置の管理に“Bツリー”（Balance Tree）を採用しているので，目的のファイルを，大容量のディスクデバイスの中から“引き当てる”のも速い．

さらにHPFSには，ファイルの実体とは別に，拡張属性（EA：Extended Attributes）と呼ばれるファイル属性をファイルごとに管理する機能がある．この拡張属性は，通常のファイル属性よりもはるかに抽象度が高い概念で，任意の属性名と任意の値を定義できる．この拡張属性が，プログラムにどのような環境を提供してくれるのかは，まだ未知数の部分も多い．しかし，複数のファイルを論理的に結合するなど，オブジェクト指向的なユーザーインターフェイスを実現する場合などに，絶大な威力を発揮する可能性がある．

OS/2 アーキテクチャ入門

入力デバイスと デバイスモニタ

PMアプリケーションでは，PMシェルがキーボードやマウスなどの入力デバイスを一括して管理しており，入力情報をメッセージの形式でアプリケーションに通知することはすでに述べた．一方のカーネルアプリケーションでキーボードやマウスからの入力情報を取得するには，まず目的のデバイスをオープンする．すると，ハンドルと呼ばれるデバイスの識別番号が返される．その後プログラムは，このハンドルを使ってデバイスにアクセスする．このようにカーネルプログラムでは，ハンドルを使用することによって，ハードウェアのポート番号や特殊な制御コードを扱うことなく，キー入力やマウスの状態を取得することが可能となっている．

DOSのシステムコールでは，スキャンコードと呼ばれる物理的なキーの識別番号によってキー入力を取得していた．これに対しOS/2のキーボードAPI（カーネルAPIの一部）では，スキャンコードだけでなく，仮想キーコードというものが返されるようになっている．この仮想キーコードを使えば，特定のハードウェアに依存することなく，キーボードを扱うことが可能となる．またOS/2では，マウス入力に関しても，マウスの物理特性に依存しない形で入力情報を取得できるようになっている．

さらに，OS/2にはデバイスモニタという概念がある．デバイスモニタは，デバイスドライバと直接通信し，その入出力データを監視したり変更したりできる機構だ．特に，キーボードデバイスと組み合わせることで効果を発揮する．これを利用すれば，DOSではキーボード割り込みを利用せざるをえなかったようなアプリケーションでも，ハードウェアに依存することなく，同様の処理を行なえる（図4）．

グラフィックデバイス

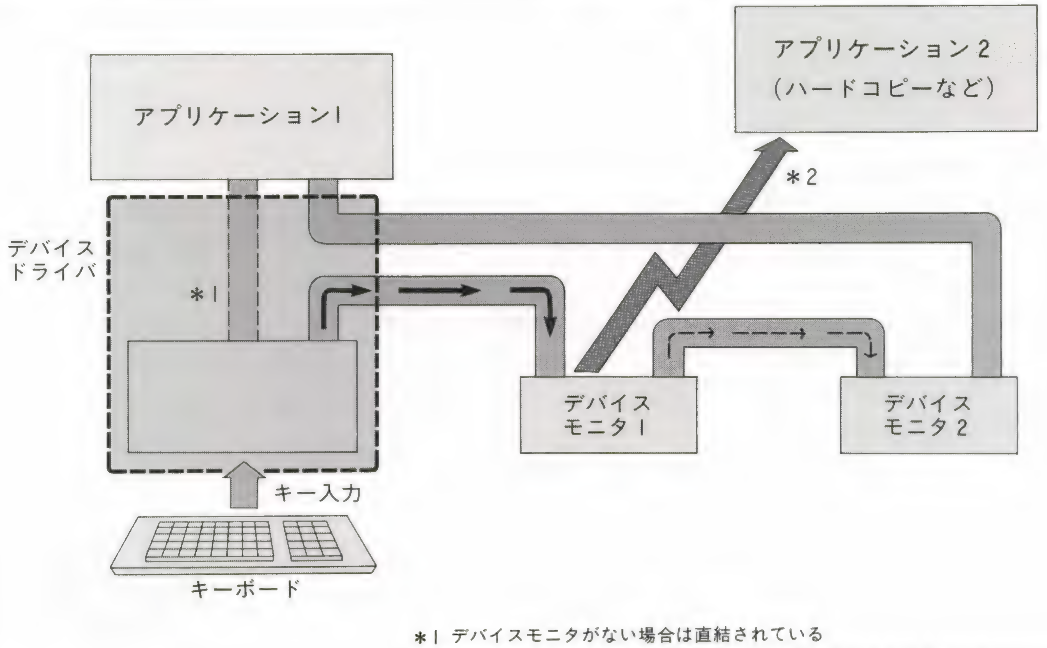
次は，プログラムからグラフィックデバイスに出力を行なう場合を考えてみよう．ひと口にグラフィックデバイスといっても，それらにはディスプレイやレーザープリンタ，プロッタなどさまざまな種類がある．そしてやっかいなことに，これらの制御方式はそれぞれ異なっている．また，ディスプレイだけを見ても，解像度や表示可能な色の数など，ハードウェアの違いによって，処理能力が大幅に異なる場合も多い．このように特性の異なるデバイスに出力するためには，個々のデバイスに応じてパラメータを計算する必要があるので，デバイスごとに処理ルーチンを用意しなければならない．従来は，この問題をアプリケーション側で処理する必要があった．

これに対しOS/2では，PMがデバイスに依存しないグラフィック環境を提供している．つまり，デバイスに依存しない，仮想的なグラフィックデバイスを利用できるのだ．PMは，“プレゼンテーション空間”（Presentation Space）と呼ばれる“仕掛け”を利用することにより，これを実現している．

●プレゼンテーション空間

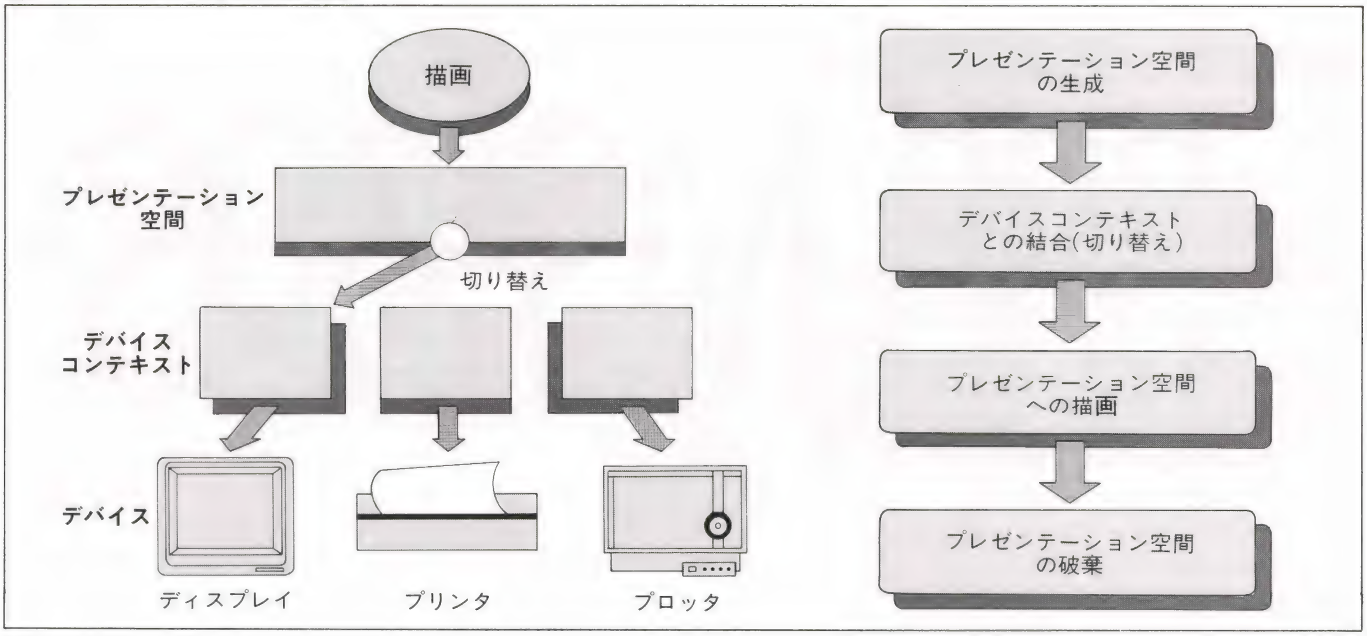
読者の多くは，C言語を使ってファイルアクセスを行なうプログラムを作成した経験があるだろう．プレゼンテーション空間（以降PSと略）は，いってみれば，

図4　デバイスドライバとデバイスモニタ



＊1　デバイスモニタがない場合は直結されている
＊2　特定のキー入力がなされた場合は，デバイスモニタ1がアプリケーション2に通知する。このとき，キー入力の情報をデバイスモニタ2に渡さないことも可能

図5　デバイスコンテキストを使った描画処理



ファイルアクセス時に使用するファイル構造体のようなものだといえる．そこで，まずはこのファイル構造体の役割について考えてみよう．

C言語の標準関数であるprintfは，ストリームに対して出力を行なう．ここで，ストリームはファイル構造体によって抽象化されたデバイスだと考えられる．printfを使って文字列をストリームに出力するとき，printfは，そのストリームデバイスがコンソールなのかファイルなのかについては何も意識しない．アプリケーションは，ファイル構造体を使うことによって，出力デバイスから独立した操作が可能となる．つまり，出力デバイスを任意長のバイト列を書き込める仮想的なデバイスとして扱えるようになる．

このファイル構造体と同じく，PSもある種の構造体データである．PSによってグラフィックデバイスは仮想化され，任意の図形要素を任意の座標位置から描画できるようになる．

従来，グラフィックデバイスの座標指定にはピクセル（画素）を用いるのが一般的だった．しかしピクセルによる座標系を扱っている限り，デバイス依存からは逃れられない．ディスプレイと高機能なレーザープリンタでは，その解像度が大きく異なるのはご存じのとおりである．PSが提供する座標系では，ピクセル単位ばかりでなく，メートル法など，日常的なスケールを使用することもできる．たとえば，mm単位の座標空間を持つPSを用いて1cmの線分を描画すれば，ディスプレイ，プリンタなど，どのような表示デバイスにも，1cmの線分が描画されるようになる．このようにPSの先にあるデバイス依存の処理は，そのすべてをシステムに一任できるため，プログラムではPSという仮想デバイスに対して描画を行ない，最終的な結果をどの物理デバイス（ディスプレイ，プロッタなど）に出力するのかを指示するだけでよい．

●デバイスコンテキスト

ファイル構造体を通したファイルアクセスでは，適切なデバイスドライバが選択され，物理的な開始位置とファイル構造体に記述されたアクセス位置（ファイルポインタの位置）に従って実際のファイルアクセスが行なわれる．このときの，物理デバイス名（デバイスドライバを決定するもの）や物理的な開始位置を記述したものに相当するのが，デバイスコンテキストである．

ファイル構造体は，その作成時に，対応するファイルのデバイスの情報と結び付けられる．これに対し，PSは，それを作成するときとは別に，デバイスコンテキストを取得することができる．この場合は，描画する前にPSとデバイスコンテキストを“結び付ける”ことによって，物理デバイスまでの経路を確保する．

PSとデバイスコンテキストを分離させたことによるメリットは大きい．PSには物理デバイスに関する情報は何もないので，描画ルーチンは，デバイスとはまったく独立に描画を行なうことができるのである（図5）．デバイスコンテキストは，PSを通じて送られたグラフィック命令を，物理デバイスドライバに対する操作に翻訳するものである．

◆

今回はアプリケーション環境という視点に立ってOS/2をながめてみた．次回は，OS/2のメモリ管理機構に焦点を当ててみようと思う．

386マシン徹底活用講座

386 de MS-DOS

連載第2回　リアルモードでdword　その2

明石真平

MS-DOSの世界では，単に高速8086としてしか使われていないことの多い386であるが，工夫しだいではMS-DOSからでも拡張された機能を引き出すことが可能である．たとえば，前回説明したように，リアルモードで32bitオペランドを使うこともできる．今回はその続きとして，やはりリアルモードで使える機能である，アドレスサイズプリフィックスとアドレッシングモード，拡張された命令について説明しよう．

拡張された アドレッシングモード

オペランドサイズプリフィックスの働きから推測すると，アドレスサイズプリフィックス(67H)を付けると今までのアドレッシングモード中の16bitレジスタがそのまま32bitに置き換わるかのように思える．たとえば8[BX+SI]にプリフィックスを付けると8[EBX+ESI]になるとか．

しかし386ではこういう拡張はしなかった．その代わりに，どのレジスタでもインデックスレジスタとして使えるように新たなアドレッシングモードを導入したのである（図1）．

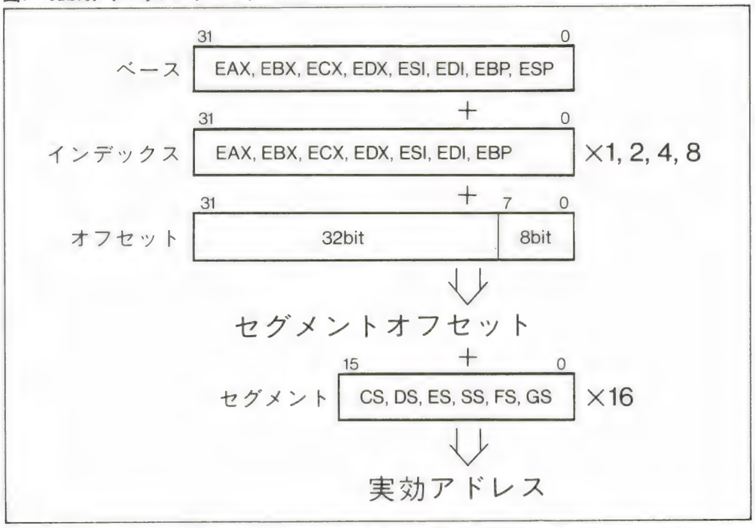
任意のレジスタをベースレジスタとし，それに任意のレジスタに1，2，4，8を掛けたものを加え（これをスケールインデックスという），さらにディスプレースメントを加えたものがセグメントオフセットとなる．3つのうちどの要素も省略可能であるが，すべてを省略することはできないし，8bitディスプレースメントだけを単独で使用することはできない（32bitディスプレースメントは単独でも使用可）．省略された各要素は0とみなされる．

ただしベースとしてEBPを使用したときはディスプレースメントを省略することはできず，8bitか32bitディスプレースメントが必要になる．また，ESPをインデックスレジスタとして使用することはできない．

もちろんこれはリアルモードや仮想8086モードでの話だから，セグメントレジスタの扱いは通常の場合とまったく同じになる．まずセグメントレジスタを16倍し，そのアドレスに上記の計算から得られるオフセットを加えたものが実効アドレスとなるのである．オフセットの計算は32bitで行なわれるが，プロテクトモードでない限り64Kbytesを超えるようなオフセットはプロセッサ例外を引き起こす（詳細は後述）．

スケールインデックスは1byte，2bytes，4bytes，8bytesの各要素からなる配列のアクセスに威力を発揮する．が，Cの構造体などでは当たり前だと思うのだが，これ以外のサイズの要素の場合はこれでは対応できない．自分で要素のサイズを掛け合わせるという，従来通りのプログラミングが必要になる．

図1　32bitアドレッシングモード



ベース，インデックス，オフセットはそれぞれ省略可能．ベースには任意のレジスタが，インデックスにはESP以外の任意のレジスタが使用可能．オフセットは8bitか32bitが使用可能．ただし，リアルモード／仮想8086モードではセグメントのオフセットは64Kbytesまでに制限される．

セグメントレジスタは，ベースとしてEBPかESPを指定した場合はSSが使われるが，それ以外のベースの場合は（ベースを省略した場合でも）DSが使われる．ただしセグメントオーバライドプリフィックスによって明示的に指示されていれば，それに従う．

386ではパイプライン処理が行なわれているので，実効アドレスの計算が他の命令の実行と並列に行なわれるときは，アドレス計算にかかる時間はみかけ上0になる．ただしベースとインデックスの両方を指定すると1クロック余分にかかる．スケールファクタの値（1，2，4，8）は実行時間には影響しない．このアドレッシングモードをアセンブラで書くときはだいたい次のようにする．

```
disp[base][index＊scale]
disp[base＋index＊scale]
[disp＋base＋index＊scale]
```

だいたいというのは，たいていのアセンブラではdispとbase, indexの順番は任意で（省略しても）構わないし，適当に［　］で囲んで＋で加算しておけば，あとは適宜判断してくれる．dispの位置も［　］の前や一番後ろでも構わない．構造体アクセスのときは“.ptrfield”のような表記を一番最後に付けることが多いだろう（この場合の“．”演算子は“＋”演算子と同じ意味である）．

しかし“[EAX][EBP]”というように，EBPを使った場合に“＊1”というスケールファクタ表記を省略すると面倒なことが起こるかもしれない．なぜなら，EAXをベースと解釈するか，EBPをベースと解釈するかでDSとSSのどちらのセグメントレジスタを使うかが変わってくるからである．

MASMやTASM, 386ASMなどでは最初のほうのレジスタをベース，後ろのほうをインデックスとして解釈するようである．だから，“[EAX][EBP]”という例では，EAXがベース，EBPがインデックスとなる．しかしIntel社のプログラマ，Ross P. Nelson著の『The 80386 Book』Microsoft Press，1988（邦訳『ザ 80386ブック』富士ソフトウエア，1989）によると，「命令がベース・レジスタとインデックス・レジスタを指定していて，そのどちらかがEBPだった場合，スケール・ファクタが存在しなければ，EBPがベースレジスタとみなされる。」と書いてある．筆者は未確認だが，ひょっとするとIntel社の開発システムに含まれるASM386ではこのようになっているのかもしれない．とすると“[EAX][EBP]”でも“[EBP][EAX]”でもEBPがベースとみなされるいうことになる．

面倒なことを避けたい人は，EBPとともに使うときはインデックスレジスタには積極的にスケールファクタを付けるようにするべきであるし（たとえ“＊1”でも），ベースレジスタを前，インデックスレジスタを後ろに書くとよいだろう．

この拡張されたアドレッシングモードは，386の特徴の1つだろう．以前と比べると，任意のレジスタをベースレジスタとして使用することができるようになったのが嬉しい．また，ライバルの（？）68020などでも使用できるスケールインデックスが装備された点も嬉しい．しかし不満をいえば，ディスプレースメントに16bitのものがないのが気になる．コードが不必要に長くなってしまう．68020などにはさらにメモリ間接アドレッシングというものがあるが，使用頻度や実現の複雑さ，実行速度などを考えると特になくても困らないだろう．

32bitアドレッシングモードの使用例を**図2**に示しておく．特に，LEA命令と組み合わせてレジスタの値を3倍，5倍，9倍するというのはスケールインデックスを備えたマシンでは定石であるので，覚えておくとよいだろう．たとえばVRAMのアドレスを計算する際によく必要になる×80倍というのは，80＝5×16であるので，まずLEA命令で5倍してから，次に4bit左シフトで16倍することによって得られる．このように，乗算する値が定数のときはうまく使うとmul命令よりも高速に計算できることがある（本当は乗算命令が高速化されればこんなことは必要ないのだが）．

図2　32bitアドレッシングモードの使用例

```
        .386
        .387
prog    segment use16
        assume  cs:prog,ds:prog,ss:prog
;
; 1次元配列のアクセス例
;
index   dw      ?
barray  db      16 dup (?)        ; byte array
warray  dw      16 dup (?)        ; word array
darray  dd      16 dup (?)        ; double word (32bit) array
qarray  dq      16 dup (?)        ; quad word (64bit) array
        movzx   edx,index
        mov     al,barray[edx]    ; 8bit int のロード
        mov     bx,warray[edx*2]  ; 16bit int のロード
        mov     ecx,darray[edx*4] ; 32bit int のロード
        fld     qarray[ecx*8]     ; 64bit real のロード
;
; 2次元配列のアクセス
;
xyarray dd      (16 * 16) dup (?)
xindex  dw      ?
yindex  dw      ?
        movzx   ebx,yindex
        shl     ebx,4+2
        movzx   ecx,xindex
        mov     eax,xyarray[ebx+ecx*4]
;
; 80x25 vram access
VRAM    db      80*25 dup (?)     ; vram の構造の定義
row     dw      ?                 ; y
col     dw      ?                 ; x
        mov     bx,row            ; bx = row*80
        lea     ebx,[ebx+ebx*4]   ; 上位 word は無視する
        shl     bx,4
        add     bx,col            ; bx = vram[row][col]
        mov     VRAM[bx],'␣'      ; set '␣'
;
prog    ends
        end
```

64Kbytesの壁

さてここまで読んできた読者は，このアドレッシングモードなら，もう壁は取り払われた，と思ったのではないであろうか？　実は，残念ながらというか，当然というか，リアルモードや仮想8086モードでは1セグメント64Kbytesの壁は依然として君臨し続けているのである．実効アドレスのオフセットが0FFFFHを超えるような場合とか，あるいはオペランドの一部が0FFFFHを超えてしまった場合，386では“一般保護例外”が起こり，INT 0DHが実行されることになっている．たとえば次のような命令を実行すると，INT 0DHが起動される．

```
mov     eax, 0123456h
mov     ebx, [eax]
```

この割り込みが起こったときの挙動は使用している機種やMS-DOS，組み込まれているドライバにより異なるので一概にはいえない．が，たいていは未定義の

割り込みベクタへ飛び込むか，ハードウェアの割り込み処理ルーチンへ飛び込むか，いずれにしろプログラムは暴走してしまうのである．

仮想8086モード下で実行している場合はそのまま仮想8086モニタ（つまりEMSの管理をしているソフト）へ飛び込んでいくが，その後の処理はどうなっているか，マニュアルにははっきりと書かれていないことが多い．たいてい「その後の動作は不安定となります」などと無責任なことが書かれている．実際には例外の起こった次の命令へ戻ってきて，何食わぬ顔をして処理を続行するものが多い．もちろん各レジスタの値はデタラメなので，「その後の動作は不安定」になってしまうようである．

こうした例外は，386のメモリ保護機構によって不正なアドレスへのアクセスが禁止されているために起こるものである．プログラムの書き方により，00H〜10Hまでのいずれの例外も発生する可能性があるが，適当な割り込みハンドラを用意しておけば，それらをトラップすることができる．これについてはいずれ連載で触れることになると思うので，今回はこれ以上は追及しないことにする．当面はそのような例外が発生しないプログラムを書くように心がけていただきたい（なんと無責任な）．

せっかくアドレスのオフセットとして32bitが使用できるのに，依然として64 Kbytesの壁が残っているとはなんとも情けない話である．この制限を取り除くためにはCPUを32bitプロテクトモードにしてやらなければならない．386のプロテクトモードは286のそれよりも拡張されており，各セグメントのリミットを4Gbytesにまですることができるようになっている．

しかしプロテクトモードにしなければならないのでは話は急に面倒になってしまうが，これは仕方がないことであろう．これについてもそのうち連載で触れるので，これ以上は追及しない．なおちょっとだけいっておくと，実はマニュアルには書かれていない386の裏ワザを使うと，リアルモードからでも4Gbytes使えたりするのである．これについてはいずれ触れることにする．

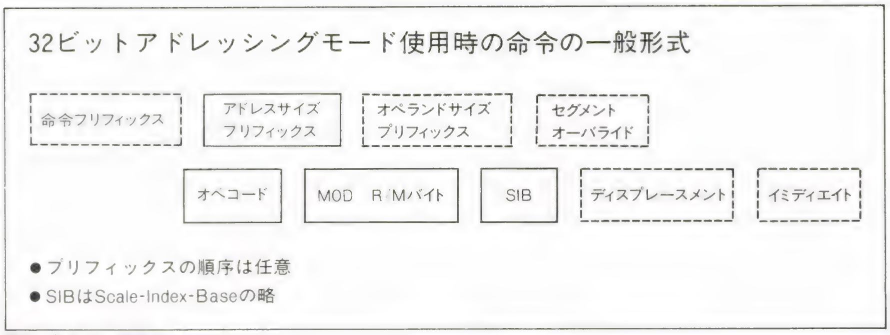


図3 命令の一般形式

オペコードの形式

32bitアドレッシングモードを使用すると，命令のコードは，**図3**のような形式になる．

アドレスサイズプリフィックスが付いた場合，今まではMOD R／Mバイトだけしかなかったところに，新たにSIB（Scale-Index-Base）バイトが増設されている．

MOD R／Mバイトの意味について簡単に説明すると，MODフィールドが11のときはR／Mフィールドがレジスタを直接表わし，00，01，10のときはR／Mフィールドと組み合わせて**表1**のようなアドレッシングモードを提供することになっていた．

しかしアドレスサイズプリフィックスが付くと，MODフィールドが11のときは今までと同様だが，00，01，10のときはR／Mフィールドと，さらに必要ならSIBバイトと組み合わせて**表2**のようなアドレッシングモードが提供されるようになったのである．

拡張された命令

それでは次に拡張された命令について見ていこう．

386では，286に比べると命令体系にいくつかの拡張が施されている．この拡張方法には2つの形式があり，1つはすでに存在していた命令に対して新しいオペランド形式を受け付けるようにしたもので，もう1つはまったく新たに命令を作ったものである．こうした拡張命令は，上手に使えばプログラムの高速化が可能になったりするので，386の機能を活かしたプログラムを書くためには，知っておいたほうがよいだろう．

表3に拡張された命令の一覧を載せておく（同時に186／286での拡張命令も記す）．ただしこれにはプロテクトモードの特権命令は含まない．それらについてはプロテクトモードの解説のときに触れることにする．

それでは，これらの命令について解説していこう．

表1 16bitモード時のアドレッシングモード

MOD R／Mバイト		実効アドレス	
MOD	R／M		
00	000	DS:［BX＋SI］	
	001	DS:［BX＋DI］	
	010	SS:［BP＋SI］	
	011	SS:［BP＋DI］	
	100	DS:［SI］	
	101	DS:［DI］	
	110	DS:disp16	
	111	DS:［BX］	
01	000	DS:［BX＋SI＋disp8］	
	001	DS:［BX＋DI＋disp8］	
	010	SS:［BP＋SI＋disp8］	
	011	SS:［BP＋DI＋disp8］	
	100	DS:［SI＋disp8］	
	101	DS:［DI＋disp8］	
	110	SS:［BP＋disp8］	
	111	DS:［BX＋disp8］	
10	000	DS:［BX＋SI＋disp16］	
	001	DS:［BX＋DI＋disp16］	
	010	SS:［BP＋SI＋disp16］	
	011	SS:［BP＋DI＋disp16］	
	100	DS:［SI＋disp16］	
	101	DS:［DI＋disp16］	
	110	SS:［BP＋disp16］	
	111	DS:［BX＋disp16］	
11		w=0	w=1
	000	AL	AX／EAX
	001	CL	CX／ECX
	010	DL	DX／EDX
	011	BL	BX／EBX
	100	AH	SP／ESP
	101	CH	BP／EBP
	110	DH	SI／ESI
	111	BH	DI／EDI

wは命令コード中のデータ長ビット

・符号拡張／ゼロ拡張指定の転送命令

符号拡張を行なう命令としては今まではCBWやCDW命令があったが，レジスタの使用に制約があった(ALもしくはAXのみ使用可能)．しかしMOVSX命令では任意のレジスタ／メモリオペランドが使用できる．

MOVZXは符号拡張ではなく，上位に0を補ってくれる．これは，従来だとレジスタをXORやSUBで0クリアしてから下位に転送するという手法で実現していた操作である．

従来の手法

```
xor       eax, eax
mov       al, 10[bp]
```

386

```
movzx     eax, byte  ptr  10[bp]
```

・ロングオフセットの条件ジャンプ命令

従来は条件ジャンプではなんと8bitオフセットしか使用できなかったのだが，これでやっと1セグメント中を自由に条件ジャンプできるようになった．use16セグメントなら16bit相対だが，use32セグメントなら32bit相対ジャンプができる．

ただしJCXZやLOOP命令(およびこれらの派生型，32bit型も含む)では相変わらず8bitオフセットの範囲でしかジャンプできない．

・汎用化された乗算命令

任意のレジスタをデスティネーションとし，ソースには任意のオペランドが使用できるようになった．

・ビット走査命令

これはビットがセットされている位置を捜すための命令である．regとr／mは共に16bit (または，共に32bit) オペランドでなければならない．

この命令はグラフィックのビットマップの操作などに役に立ちそうに思える．しかし，第2オペランドの内容が0のときは，実行クロック数は10クロックだが，それ以外のときは1bit走査するごとにさらに3クロックずつかかるので，dwordのときは最悪100クロック程度かかる．だから非常に速度が要求されるときはもっと別の高速化の手段をとったほうがいいかもしれない．そうでなければまあまあ役に立ちそうだ．

```
mov   eax, 00008000h
bsf   ecx, eax   ;  ecxは15になる
bsr   edx, ecx   ;  edxは3になる
```

・ビット操作命令

これは68020などで使えるビットフィールドの操作命令ではないことに注意されたい．単なる1bitのテスト／操作命令である．こんなものはモトローラでは68000のときから使えたのだが．まぁないよりはましか．

```
mov     eax, 0000ffffh
bt      eax, 20  ;  Cflag＝0
btc     eax, 7   ;  Cflag＝1
                 ;  eaxは0000ff7fhになる
btr     eax, 15  ;  Cflag＝1
                 ;  eaxは00007f7fhになる
bts     eax, 31  ;  Cflag＝0
                 ;  eaxは80007f7fhになる
```

・倍長シフト命令

これは2つのオペランド (r／mとreg)をつないで左(SHLD)もしくは右(SHRD)へimm8／clbitシフトするための命令である．ただし第2オペランドの内容は変わらず，第1オペランドのみが変更を受ける(**図4**)．シフトされて入ってくるデータが第2オペランドで，出ていくビットは捨てられる．

この命令を使うと，通常のシフトでは簡単にできない4bytes以上からなる多倍長データのシフトが行なえる．今までは，1オペランドに入りきらないデータのシフトはRCLやRCR命令で1bitずつシフトするか，それ以上の場合はandやorを組み合わせてシフトを実現するしかなかった．

これは他のシフト命令にもいえることだが，386ではバレルシフタと呼ばれる，一度に任意ビット数のシフトをすることができる機構を備えているので，この命令はシフト量にかかわらず3／7クロックで実行することができる．バレルシフタはチップ上で結構面積を占有するので，あまり速度を追求しないプロセッサや面積に制約のあるプロセッサでは装備されていない．

表2　32bitモード時のアドレッシングモード

MOD R／Mバイト		実効アドレス	SIB併用時の実効アドレス(SIB=ss:iii:bbb)	
MOD	R／M		bbb	実効アドレス
00	000	DS:[EAX]	000	DS:[EAX＋(Scale-Index)]
	001	DS:[ECX]	001	DS:[ECX＋(Scale-Index)]
	010	DS:[EDX]	010	DS:[EDX＋(Scale-Index)]
	011	DS:[EBX]	011	DS:[EBX＋(Scale-Index)]
	100	SIB併用を示す	100	SS:[ESP＋(Scale-Index)]
	101	DS:disp32	101	DS:[disp32＋(Scale-Index)]
	110	DS:[ESI]	110	DS:[ESI＋(Scale-Index)]
	111	DS:[EDI]	111	DS:[EDI＋(Scale-Index)]
01	000	DS:[EAX＋disp8]	000	DS:[EAX＋(Scale-Index)＋disp8]
	001	DS:[ECX＋disp8]	001	DS:[ECX＋(Scale-Index)＋disp8]
	010	DS:[EDX＋disp8]	010	DS:[EDX＋(Scale-Index)＋disp8]
	011	DS:[EBX＋disp8]	011	DS:[EBX＋(Scale-Index)＋disp8]
	100	SIB併用を示す	100	SS:[ESP＋(Scale-Index)＋disp8]
	101	SS:[EBP＋disp8]	101	SS:[EBP＋(Scale-Index)＋disp8]
	110	DS:[ESI＋disp8]	110	DS:[ESI＋(Scale-Index)＋disp8]
	111	DS:[EDI＋disp8]	111	DS:[EDI＋(Scale-Index)＋disp8]
10	000	DS:[EAX＋disp32]	000	DS:[EAX＋(Scale-Index)＋disp32]
	001	DS:[ECX＋disp32]	001	DS:[ECX＋(Scale-Index)＋disp32]
	010	DS:[EDX＋disp32]	010	DS:[EDX＋(Scale-Index)＋disp32]
	011	DS:[EBX＋disp32]	011	DS:[EBX＋(Scale-Index)＋disp32]
	100	SIB併用を示す	100	SS:[ESP＋(Scale-Index)＋disp32]
	101	SS:[EBP＋disp32]	101	SS:[EBP＋(Scale-Index)＋disp32]
	110	DS:[ESI＋disp32]	110	DS:[ESI＋(Scale-Index)＋disp32]
	111	DS:[EDI＋disp32]	111	DS:[EDI＋(Scale-Index)＋disp32]

MOD	R／M	w=0	w=1
11	000	AL	AX／EAX
	001	CL	CX／ECX
	010	DL	DX／EDX
	011	BL	BX／EBX
	100	AH	SP／ESP
	101	CH	BP／EBP
	110	DH	SI／ESI
	111	BH	DI／EDI

wは命令コード中のデータ長ビット

Scale-Indexの指示

ss	スケールファクタ	iii	インデックスレジスタ
00	＊1	000	EAX
01	＊2	001	ECX
10	＊4	010	EDX
11	＊8	011	EBX
		100	NO INDEX
		101	EBP
		110	ESI
		111	EDI

表3　追加／拡張された命令

■186／286で追加された命令

ニーモニックの形式		意味
PUSH	imm	イミディエイト値のpush
PUSHA		全レジスタのpush／pop
POPA		
IMUL	reg,r／m,imm	定数の乗算
Shift	r／m,imm8	2ビット以上の定数シフト
INS		繰り返しI／O命令
OUTS		
ENTER		スタックフレームの管理
LEAVE		
BOUND		範囲チェック

■386で追加された命令

・符号拡張／ゼロ拡張指定の転送命令

ニーモニックの形式		クロック	意味
MOVSX	reg16,r／m8	3／6	バイト→ワードの符号拡張転送
MOVSX	reg32,r／m8	3／6	バイト→ダブルワードの符号拡張転送
MOVSX	reg32,r／m16	3／6	ワード→ダブルワードの符号拡張転送
MOVZX	reg16,r／m8	3／6	バイト→ワードのゼロ拡張転送
MOVZX	reg32,r／m8	3／6	バイト→ダブルワードのゼロ拡張転送
MOVZX	reg32,r／m16	3／6	ワード→ダブルワードのゼロ転送

・ロングオフセットの条件ジャンプ命令

ニーモニックの形式		クロック	意味
Jcc	rel16／rel32	7＋m／3	16／32ビットオフセットの条件ジャンプ

・汎用化された乗算命令

ニーモニックの形式		クロック	意味
IMUL	reg,r／m	9～41	reg ← reg×r／m

・ビット走査命令

ニーモニックの形式		クロック	意味
BSF	reg,r／m	10＋3n	1のビットの検索（昇順検索）
BSR	reg,r／m	10＋3n	1のビットの検索（降順検索）

・ビット操作命令

ニーモニックの形式		クロック	意味
BT	r／m,reg	3／12	ビットテスト
BT	r／m,imm8	3／6	
BTC	r／m,reg	6／13	ビットテスト＆反転
BTC	r／m,imm8	6／8	
BTR	r／m,reg	6／13	ビットテスト＆リセット
BTR	r／m,imm8	6／8	
BTS	r／m,reg	6／13	ビットテスト＆セット
BTS	r／m,imm8	6／8	

・倍長シフト命令

ニーモニックの形式		クロック	意味
SHLD	r／m,reg,imm8	3／7	r／mとregを連結して左シフト
SHLD	r／m,reg,cl	3／7	
SHRD	r／m,reg,imm8	3／7	r／mとregを連結して右シフト
SHRD	r／m,reg,cl	3／7	

・条件セット命令

ニーモニックの形式		クロック	意味
SETcc	r／m8	4／5	条件成立なら1，不成立なら0をセット

・FS，GSオーバライト

FS:
GS:

・セグメントレジスタへのロード命令

ニーモニックの形式		クロック	意味
LFS	reg,mem	7,p=20	FSとregへロード
LGS	reg,mem	7,p=20	GSとregへロード
LSS	reg,mem	7,p=20	SSとregへロード

・特殊用途用レジスタとの転送命令

ニーモニックの形式		クロック	意味
MOV	r32,CRn	6	制御レジスタCRnとの転送命令
MOV	CRn,r32	4～10	
MOV	r32,DRn	14～22	デバッグレジスタDRnとの転送命令
MOV	DRn,r32	14～22	
MOV	r32,TRn	12	テストレジスタTRnとの転送命令
MOV	TRn,r32	12	

命令の後の 3／6 などは命令の実行クロック数を表わす．／の前がオペランドがレジスタの場合で，後ろがメモリ中のオペランドの場合を表わす．この中には命令のフェッチやデコード，実行アドレスの計算に要する時間は含まないし，メモリのウエイトはなく，メモリオペランドはアラインされているものとする．10＋3nは1ビットごとに3クロックずつ増加することを，p=20はプロテクトモードでの実行時間を表わす．regはレジスタオペランド，r／mはレジスタオペランドかメモリオペランド，memはメモリオペランド，immはイミディエイトデータをそれぞれ表わす．

・条件セット命令

これはJccと同様に，指定された条件が成立しているかどうかを調べる命令だが，条件の成立／不成立に応じて，ジャンプするのではなく，1／0を指定された8bitオペランドに格納する．C言語の条件演算式に対応するものと考えてもらえばよいであろう．680x0などでは0／1ではなく，00H／FFHという値がセットされるようになっているが，これはPascalなどの処理系の多くがTrue＝－1, False＝0としていたのを反映した結果であろうか．386のほうが（少しだけ）C言語向きである．

```
cmp   eax, ebx
seta  al  ;  AL=(EAX > EBX)
setb  bl  ;  BL=(EAX < EBX)
sete  cl  ;  CL=(EAX==EBX) ?  1:0
neg   cl  ;  CL=(EAX==EBX) ?-1:0
```

・セグメントレジスタへのロード命令

これもセグメントレジスタの増設に伴って必然的に作られた命令である（といってもSSは前からあったがLSS命令はなかった）．

・特殊用途用レジスタとの転送命令

これもレジスタの増設に応じて作られたものだが，詳細についてはプロテクトモードの説明の回に譲ることにする．

サンプルプログラム

それでは今回の復習を兼ねてサンプルプログラムを作成してみよう．

前回のbignum演算ルーチンでは単に32bit加算命令しか使用していなかったので，今回は386の拡張された命令を使ってもっと386らしく書く．そして，前回やり残した部分についても手直ししていくことにする．

主な拡張点として，まず結果用の領域をこのルーチンが自動的に割り当てるようにしたことがある．最初に必要な大きさを（1つ大きめに）見積り，malloc( )関数を呼び出して結果の領域を割り当てている．そして両方のオペランドの長さが異なる場合でも正しく加算を行なえるようにした．また，結果の最上位のdwordが0のときは，その部分を切り詰めている．これは最初の見積りが多すぎた場合

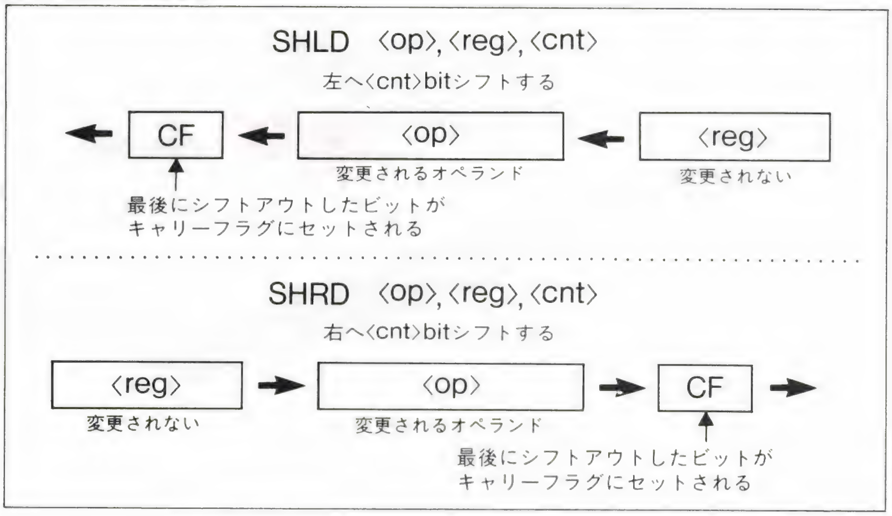
の処理である.

プログラムでは，まず最初に2つのデータの長さを調べ，“長いほうのデータの長さ＋1”桁分のbignum用データ領域を用意している．これにデータ長を示す1dword分の大きさを加え，その後malloc( )でデータ領域を確保している．もしmalloc( )できなければDX：AXを0 (NULL) にして戻る．

計算は，まず最初にデータを下位のほうから加えていくことから始める（加える桁数は短いほうのデータに合わせる）．次に長いほうのデータに対して桁上げ計算を行なう．そして最後にMSD (Most Significant Dword)からの桁上げを処理し，MSDが0なら結果の長さを1減らしている．

サンプル全体としてみると，残念ながらあまり386らしくないところもあるが，

図4　SHLD, SHRD命令



強いてあげるとすると，3つのセグメントレジスタを使ったポインタ操作，32bit演算などが386のプログラムらしいといえるだろうか．

このプログラムではまだ負のデータや他の種類の演算を扱うことができないし，64Kbytes以上のデータも扱うことができないが，いずれ改良する予定である．

サンプルプログラム

```
                page    , 131
;* NAME: BigComp.asm ver 1.1
;* DESC: add two BIGNUMs for MS-C ver 5.1/6.0
;* ASM:  use MASM ver 5.1,  TASM ver 1.0 or later
                .model  large
                .386
                .code
                extrn   _malloc:far
_ARG0           =       6                       ; param 0
_ARG1           =       10                      ; param 1
;+ struct { long Length;
;+          long NumDigit[Length]; }
;* BigNum far *BigAdd(
;*      BigNum far *src1,   ==> fs:si
;*      BigNum far *src2)   ==> gs:bx
;* return ==> NULL : error
                align   4
                public  _BigAdd
_BigAdd         proc    far
                push    bp                      ; func prologue
                mov     bp, sp
                push    si                      ; save reg var
                push    di
                lfs     si, _ARG0[bp]           ; src1 ptr
                lgs     bx, _ARG1[bp]           ; src2 ptr
                mov     ax, fs:[si]             ; src1 length
                cmp     ax, gs:[bx]             ; src2 length
                jae     short _BA1              ; src1 is longer or same?
                mov     ax, gs:[bx]             ; no,  src2 is longer
                lfs     si, _ARG1[bp]           ; swap src1 & src2
                lgs     bx, _ARG0[bp]
_BA1:           inc     ax
                push    ax                      ; save length (-6[bp])
                lea     ax, 4[eax*4]            ; cnv to dword cnt
                push    fs
                push    si                      ; -10[bp]
                push    gs
                push    bx                      ; -14[bp]
                push    ax                      ; push param
                call    _malloc
                or      dx, dx                  ; error ?
                je      short _BAEnd            ; yes
                mov     es, dx
                mov     di, ax
                push    dx                      ; result ptr
                push    ax
                lfs     si, -10[bp]
                lgs     bx, -14[bp]
                movzx   ecx, word ptr -6[bp]
                mov     es:[di], ecx
                mov     cx, gs:[bx]             ; length
                add     si, 4                   ; src1 (long)
                add     bx, 4                   ; src2 (short)
                add     di, 4                   ; result
                clc
                jcxz    BigAddLoop1
BigAddLoop:     lods    dword ptr fs:[si]       ; add common part
                adc     eax, gs:[bx]
                stosd
                lea     bx, 4[bx]
                loop    BigAddLoop
BigAddLoop1:    pushf                           ; add upper part
                mov     cx, -6[bp]
                mov     bx, -14[bp]
                sub     cx, gs:[bx]
                dec     cx                      ; same length ?
                je      short _BATruncate       ; yes
                popf                            ; no,  src1 is longer
BigAddLoop2:    lods    dword ptr fs:[si]       ; adjust upper digits
                adc     eax, 0
                stosd
                loop    BigAddLoop2
                pushf
; compute most significant dword (MSD)
_BATruncate:    popf
                mov     eax, 0
                adc     eax, 0
                movzx   eax, al                 ; eax := contents of MSD
                mov     es:[di], eax
                mov     dx, es
                movzx   eax, word ptr -20[bp]
; truncate MSD if 0
                jne     short _BAEnd            ; MSD is 0 ?
                dec     dword ptr es:[eax]      ; yes
_BAEnd:         lea     sp, -4[bp]              ; epilogue
                pop     di
                pop     si
                leave
                ret
_BigAdd         endp

                end
```

入門システムプログラム

抽象化とawkによるプロトタイプ作成

画面書き換えの最適化

国際日本文化研究センター(情報学)
小野芳彦

前回はプログラミングの精神訓話が多く，プログラムのアルゴリズムという点では，多少物足りなかったかもしれない．そこで，今回は少し賢いプログラムを設計してみよう．具体的には，前回構築した抽象的なデータ構造の取り扱い方の例として，画面制御ライブラリを画面の書き換えを必要最小限に抑える方向で最適化してみたい．併せて，必要なアルゴリズムやデータの扱い方のノウハウについても触れていくつもりである．

前回の復習

前回は，画面制御の抽象化というテーマで，いろいろなアルゴリズムをライブラリで実現した．ライブラリは，メインプログラムから共通に使えるようにインターフェイスを統一したが，awkには抽象データ型の考え方が取り入れられていないので，多少テクニカルなコーディングを行なっている．前回の説明が舌足らずで分かりにくい面があったので，まずは補足を兼ねてライブラリの復習から始めよう．ただし，前回とまったく同じライブラリを載せるのも芸がないので，今回の新しい話題に適した第4のライブラリを載せることにする．どういったところが適しているかは，おいおい分かってくるであろう（**リスト1**）．

まず，このライブラリを利用するうえでメインのawkプログラムとリンクをとる必要がある．Cならば，ソースプログラムの上でのリンクは，#includeというプリプロセッサへのコマンドという形で取り込むファイルを指定する．(j) gawkではプログラムファイルを指定する-fオプションが複数指定できるので，

jgawk -f main.awk -f lib1.sub

のように指定する．

両者のインターフェイスは，Cと同様に関数の引数として渡すか，大域的変数として暗黙のうちに渡すかの2通りの方法がある．awkの普通の変数はBASICのようにすべてが大域変数なので，名前の衝突には多大の注意を払わなければならない．関数を定義できるようにawkの言語仕様を拡張したとき，バグを出しにくくするといった配慮が欠けていたようであり，残念である．

ただBASICとは異なり，局所変数を宣言することはできる．前回は説明が必要でないようにすべて大域変数で局所変数的な作業も行なってきたが，今回のように再帰呼び出しを利用する場合(replacestr( ))はどうしても必要である．局所変数は，奇妙なことに関数の引数宣言の続きに単にカンマで区切って書くことになっている．引数が局所変数としてスタック上にとられることを利用して，このように書くのであろうが，どうも慣れないと落ち着かない．ここでは，ここだけの慣例として，本当の引数と局所変数の区切りに目印としてスペースを空けることにしよう（emptyscreen( )などの宣言部を参照）．

インターフェイスを設計するうえでのもう1つのawkの欠点は，Cの構造体のようなデータをまとめて1つの変数とする機能がないことである．ここではawkの配列の特徴を利用して，構造体もどきを作って利用することにした．**リスト1**では，画面を表わす構造として，コラム数，行数，X座標，Y座標，行の配列の5つの要素をscreenという1つの引数で引渡す必要がある．これを実現するのに，screenを配列とし，行の配列以外はメンバー名を文字列リテラルとするインデックスを与えるという方式をとった（編集部から示唆をいただいた．感謝）．これこそawkの醍醐味であるが，いつでも使えるというものではない．たとえば，配列が2つ以上含まれていれば，多少工夫をこらさなければならない．

これでも，違った擬似構造体が渡された場合，まったくチェックできないという問題が生じる．しかし，ライブラリを作るのにはかなり役に立つ方法である．

もう1つの問題は，このように配列を

UNIXの画面制御

UNIXでは，画面の制御のためにいくつかの方法を用意している．中でも代表的なのはtermcapである．termcapは，各種の端末の制御情報を記述したデータベースであり，画面制御の機能と実際の端末を制御するためのシーケンスが対応づけられている．termcapを参照すれば，アプリケーションプログラムを作成する際には端末の種類を意識することなく，最適な制御を行なうことができる．初期の頃のネットワークでは通信速度が遅かったこともあって，ホストと端末の間がの通信量を極力減らす方向で最適化する必要があった．こうした場合にも対応できるように，termcapには低レベルの制御を行なうためのシーケンスなども豊富に登録されている．そのため，アプリケーション側で目的に応じた最適化を工夫することも可能である．

構造体に使ったために，呼ぶ側で配列の要素を引数に与えることができないという問題が生じる．たとえば，screenの配列を作り，その要素の1つを引数にしてこのライブラリを呼び出してもうまくは動かない．これについては，それなりの対処が個々に必要であるが，ここでは触れないことにしよう．

リスト1

```
# Data Type and Procedures
function emptyscreen(screen, cols, lines,  i) {
      screen["COL"] = cols+0
      screen["LINE"] = lines+0
      screen["XPOS"] = screen["YPOS"] = 1
      for(i = 1; i < screen["LINE"]; i++) {
            screen[i] = ""
      }
}
function move(screen, col, lin) {
      if(lin+0 > screen["LINE"]) lin = screen["LINE"]
      if(col+0 > screen["COL"]) col = screen["COL"]
      screen["XPOS"] = col
      screen["YPOS"] = lin
}
function nextline(screen) {
      screen["XPOS"] = 1
      screen["YPOS"]++
      checkscrollup(screen)
}
function checkscrollup(screen) {
      while(screen["YPOS"] > screen["LINE"]) {
            deleteline(screen, 1)
      }
}
function deleteline(screen, line) {
      for(; line+0 < screen["LINE"]; line++) screen[line+0] = screen[line+1]
      screen[line] = ""
}
function replaceline(screen, str,  s) {
      s = screen[screen["YPOS"]]
      while(length(s) < screen["XPOS"] - 1) s = s "␣"
      s = substr(s, 1, screen["XPOS"] - 1) str substr(s, screen["XPOS"]+length(str))
      if(length(l)) > screen["COL"]) {
            screen[screen["YPOS"]] = substr(l, 1, screen["COL"])
            screen["YPOS"]++; screen["XPOS"] = 1
            replaceline(screen, substr(l, screen["COL"]+1))
      }
      else {
            screen[screen["YPOS"]] = l
            screen["XPOS"] += length(str)
      }
}
function redrawscreen(screen,  r, i) {
      r = ""
      for(i = 1; i <= screen["LINE"]; i++) {
            r = r screen[i]
            if(length(screen[i]) < screen["COL"]) r = r "\n"
      }
      return r
}
function redrawline(screen, lineno) {

      return screen[lineno]
}
```

今回のテーマ

今回のテーマは，「画面書き換えの最適化」である．前回は，画面を書き換えるのに，いったん画面をクリアして，次の画面を書いていた．これを，以下のように変える．まず，次の画面を一度に用意する．そして，前の画面と次の画面を全体として比較し，残せる部分は最大限残すようにして，画面を書き換えようというのである．このとき，前回説明した，現実の画面制御を思い出せば，どこが共通しているかを調べてやる必要があることが分かる．

画面を折り畳まれた1次元の文字列であるとみなす場合に，実際の文字単位の削除や挿入が画面の端を越えないようになっていることは，大きな障害の1つである．もちろん，それを手当てすることは可能であるが，繁雑であることは否めない．また，文字単位の削除や挿入機能を備えていない端末も多い（PC-9801のMS-DOSが用意する画面制御機能にもない）．挿入や削除がなければやれることは限られ，プログラムを考える面白みが半減してしまうだろう．そこで，もうすこし単位の大きい「行」を考えよう．つまり，行ごとに一致するかどうかの比較を行なうようにするのである．うまいことに，行の削除と挿入をサポートするシステムは多いから，これは有効であるに違いない．

基本のアルゴリズム

まず，準備として，最大の共通行数を計算するアルゴリズムから始めよう．この問題には，LCS (Longest Common Subsequence) という名前が付けられ，解のアルゴリズムについてたくさんの議論がなされてきている．LCSとは，2つの文字列に共通で，かつ最長の部分列のことである．部分列とは，もとの列から，要素を出現順序を変えずに取り出したもので，要素は必ずしも連続していなくてもよい（次ページ図1）．

こういう問題では，分割統治の手法が役立つことが多い．分割統治法とは，問題を規模の小さい問題に還元して解くような方法全般に名付けられたものである．

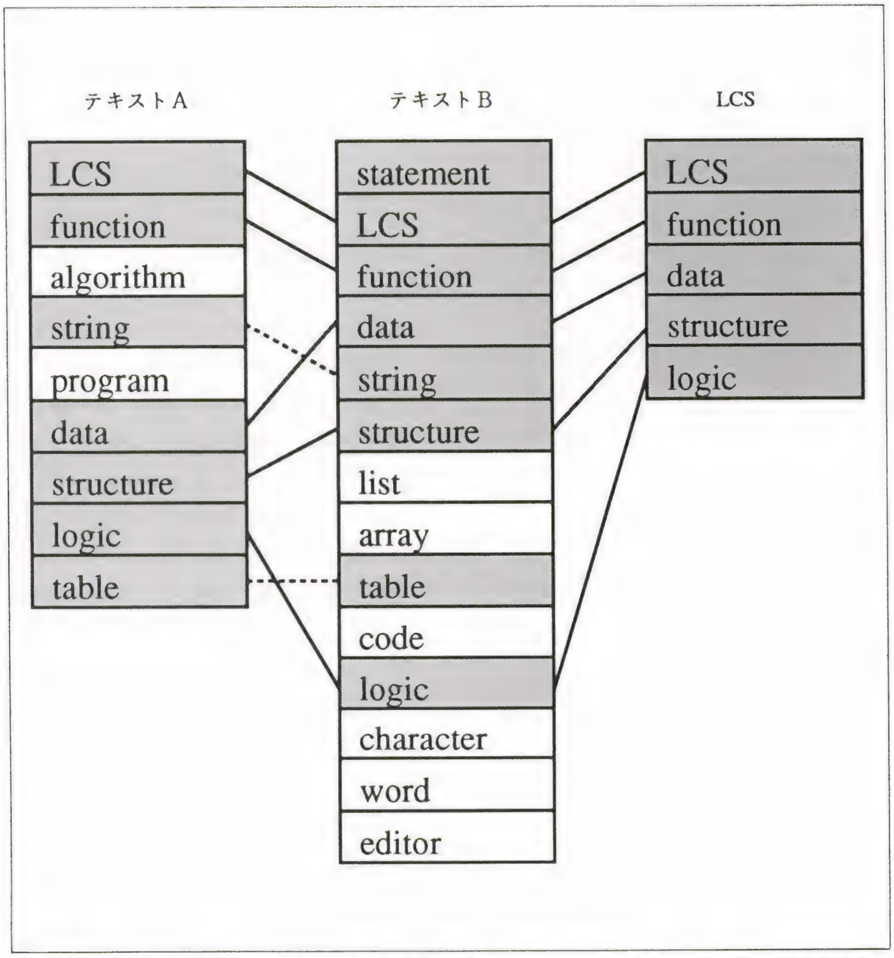


図1　この例の場合，テキストAとテキストBの要素（単語）のうち，両者に共通なものを取り出して出現順序を変えないようにして並べたものがLCSとなる．“data”の代わりに“string”を選ぶこともできるが，LCSの長さは同じである．“table”についても同様である．

リスト2

```
function LCS(a, b,  lasta, heada, lastb, headb, temp1, temp2) {
     if(a == "") return 0
     if(b == "") return 0
#
     lasta = jsubstr(a, jlength(a), 1)
     heada = jsubstr(a, 1, jlength(a)-1)
     lastb = jsubstr(b, jlength(b), 1)
     headb = jsubstr(b, 1, jlength(b)-1)
#
     if(lasta == lastb) return LCS(heada, headb)+1
#
     temp1 = LCS(a, headb)
     temp2 = LCS(heada, b)
#
     if(temp1 > temp2) return temp1
     else return temp2
}
function LCSinit() {
# 何もしない
}
```

規模が小さい問題に還元するには，次のように考える．比べる列を

$a_1\ a_2\ a_3\ \cdots\ a_{m-1}\ am$

$b_1\ b_2\ b_3\ \cdots\ b_{n-1}\ bn$

と書くことにする．a_mとb_nが一致する場合，それぞれ1ずつ短くした

$a_1\ a_2\ a_3\ \cdots\ a_{m-1}$

$b_1\ b_2\ b_3\ \cdots\ b_{n-1}$

の最大共通長LCS(m−1,n−1)を求め，それに1を加えたものが求めるLCS(m,n)になることは，すぐ理解できる．さらに，一致しない場合はaを1短くしたLCS(m−1,n)とbを1短くしたLCS(m,n−1)を求め，そのうちの大きいほうをとればよいことも分かる．これを再帰的に適用していけば，LCS( )の2つの引数は小さくなっていき，どちらかが0になる．LCS(0,y)やLCS(x,0)は，空列と部分列の共通部分の最大長を計算せよということを意味し，これは当然0である．ここで値が求まるから，順次呼ばれたLCS( )に戻っていき，最終的に最大共通長が求まる．

さて，これを正直にコーディングすると**リスト2**のようになる．ここでは2つの引数は文字列を渡すというawk向きのデータ型にしている．また，再帰呼び出しを行なうので，関数の中で使う変数は，局所変数として引数宣言の後半に書いておかなければならない．

これを実際に動かしてみればすぐ分かることだが，調べる2つの列が長いと実行にえらく時間がかかる．この関数の呼び出し回数はいったいどのくらいになるであろうか．両方の列の長さをmとし，最悪のケースとして一致するところが1つもないとする．LCS(m,m)はLCS(m,m−1)とLCS(m−1,m)を呼び出し，さらにそれらが2つずつLCS( )を呼び出すから，ほぼ2のm乗に比例するだけの回数の呼び出しが必要になってしまう．本番で必要なm＝画面の行数24を代入してみれば，大変な大きさになっていることが分かる．何かもっとうまい方法にしなければならない．

もう少し詳しく見ていくと，改良の余地が見えてくる．LCS(m,m−1)が呼び出すのはLCS(m−1,m−1)とLCS(m,m−

2)でありLCS(m−1,m)が呼び出すのはLCS(m−2,m)とLCS(m−1,m−1)である．ここで，LCS(m−1,m−1)が2度呼び出されている．LCS(x,y)全体としてxとyの組み合わせはmの２乗個しかないのであるから，同じ引数の呼び出しを２度目にはしないように工夫をすれば，mの２乗回ですむはずである．

すぐに思い付くawkらしい解決法をここで使ってみよう．つまり，１度LCS( )を計算したら，その値を覚えておくのである．呼び出す前に値が定義されているかどうか調べ，されていればそれが使える．awkの場合，inという演算子でインデックスが示すところに値が代入されているかどうかを調べることができる（以前のプログラムでもそれを使っていた）．それを組み込むと**リスト３**のようになる．ここで注意しなければならないのは，値を覚えておくための配列は，次に使うときのために値を消去しておかなければならない，つまり，in演算子で調べたとき値が定義されていない状態にしておかなければならないのである．そのためには，delete文で配列の要素を消すという特別の操作をしなければならない．この操作は，すべてのインデックスに対応させて行なわなければならないが，簡単にfor文と組み合わせて行なう方法がある（**リスト３**のLCSinit( )を参照）．実際のインデックスは２次元であるが，gawkの配列の実現方式により，１次元と同様に扱ってもかまわないことになっている．今回は以下のようにインデックスを２重に回せば連続して消去できるケースなのであまりありがたみはないが，インデックスが飛び飛びのケースでは非常に有効な方法なので覚えておかれるのがよかろう．

```
for(i=1; i<=SIZEA; i++){
    for(j=1; j<=SIZEB; j++){
        delete lcsvalue [i, j]
    }
}
```

改良したアルゴリズム

実は，この方法は多少寄り道であって，本筋はもっと直接的な，awkに頼らない方法である．本筋に戻ろう．**リスト２**や**リスト３**は，いわゆるトップダウンのアプローチであった．これをボトムアップのアプローチに変更すると，**リスト２**でみられたような計算時間の爆発を避けることができることがある．今回はそのような例に当たる．

リスト３

```
function LCS(a, b,  lasta, heada, lastb, headb, temp) {
    if(a == "") return 0
    if(b == "") return 0
    if(!((a, b) in lcsvalue)) {
        lasta = jsubstr(a, jlength(a), 1)
        heada = jsubstr(a, 1, jlength(a)-1)
        lastb = jsubstr(b, jlength(b), 1)
        headb = jsubstr(b, 1, jlength(b)-1)
        if(lasta == lastb) {
            lcsvalue[a, b] = LCS(heada, headb)+1
        }
        else {
            temp = LCS(a, headb)
            lcsvalue[a, b] = LCS(heada, b)
            if(temp > lcsvalue[a, b]) lcsvalue[a, b] = temp
        }
    }
    return lcsvalue[a, b]
}
function LCSinit() {
    for(i in lcsvalue) delete lcsvalue[i]
}
```

ここで求めなければならない値はLCS(m,n)だけだが，今まで見てきたように，必要になるのはLCS(＊,＊)の全体である．これをボトムアップ方式で配列（lcs[　]）の上に作っていこう（**リスト４**の前半部分）．インデックスの若いほうからlcs[i,j]を計算する．行を比較して等しい場合には，それぞれのインデックスが１つ前のlcs[i−1,j−1]に１を加えた数をlcs[i,j]に格納し，等しくない場合には，lcs[i,j−1]とlcs[i−1,j]の大きいほうをlcs[i,j]に格納すればよい．初めは，iやjが１であるから，lcs[0,0]のような未定義の値を取り出そうとするが，awkの規定でそれらは空文字列になり，さらにそれらに数を加えると空文字列を0とみなす規定からうまく求める数になってくれる．このようにして，配列の初期値を設定することを省略できるのである．が，これはいつでもうまくいくとは限らないので注意を要する．両インデックスが最大となるlcs[　]の値が，求める値である．

さて，この最長値を求めるのが目的であれば，実は２次元の配列にしなくても，１次元の配列で十分である（これは考えればすぐに分かる）．しかし，目的は最長の部分列を具体的に指摘することである．それにはどうすればいいかを，次に考えよう．まず，具体例からとりかかる．下記の２つ場合の文字列のlcs[　]を書いてみよう．

(1)

	a	b	a	b	a	b
b	0	①	1	①	1	①
a	①	1	②	2	②	2
b	1	②	2	③	3	③
a	①	2	③	3	④	4
b	1	②	3	④	4	⑤
c	1	2	3	4	4	5

(2)

	a	b	a	b	a	c
b	0	①	1	①	1	1
a	①	1	②	2	②	2
b	1	②	2	③	3	3
a	①	2	③	3	④	4
b	1	②	3	④	4	4
a	①	2	③	4	⑤	5

結果は，両者ともに共通部分列の長さは５である．両表の上に比較の結果一致した場所を丸付きで示してある．このうちのどの部分を実際に共通だとすれば最長になるかを見てとってほしい．(1)の場合はbababであり，(2)の場合はababa

である．その場所だけを取り上げて丸を付けてみると，

```
        (1')                      (2')
 | a b a b a b              | a b a b a c
-+-------------            -+-------------
b| 0 ① 1 1 1 1             b| 0 1 1 1 1 1
a| 1 1 ② 2 2 2             a| ① 1 2 2 2 2
b| 1 2 2 ③ 3 3             b| 1 ② 2 3 3 3
a| 1 2 3 3 ④ 4             a| 1 2 ③ 3 4 4
b| 1 2 3 4 4 ⑤             b| 1 2 3 ④ 4 4
c| 1 2 3 4 4 5             a| 1 2 3 4 ⑤ 5
```

である．これらの場所の特徴を見ると，次のことが見えてくる．これらは皆，同じ列や段にはない．対応は1対1であるから当然のことである．また，同じ数をつないで1つのエリアとみるとき，これらがすべてそのエリアの左上角にある．つまり，そこは先頭から見て一番左の一致点になっているわけである．もちろん，そのほかに一致する点があって，それを採用しても最長部分列が得られる場合もあるが，とにかく結果が1つ得られればよいのである．別のケースをわざわざ探す必要はない．最長の共通部分列は，大きい数字のエリアから始めて，そのエリアの左上角を見つけ，次にその地点より左上で，次に大きい数字のエリアの左上角を探すというふうに全体を走査すれば見つけられる．

これをプログラムに書けば**リスト4**の後半になる．上や左に行けなくなった地点が角である．まず上，次に左（あるいはその逆順）にどんどん進んでいくという方法もあるが，ここでは対角線上にまず進むという方針をとった．理由は次の節で説明する．プログラムでは，隣の値とは差が1以下であることを暗黙のうちに利用している．

さて，これで最長部分列は求められた．残る問題は，横軸の列から縦軸の列へ変更するのにどのように変更するかである．lcs[ ]をたどる際に，同じ数字のエリアを左にたどった場合，その元の位置の横軸側の文字は一致したとみなされなかったことになる．この文字は横軸側から縦軸側に変更する際に不要となるのであるから，削除する必要がある．次に，同じ数字のエリアを上にたどった場合，その元の位置の縦軸側の文字は一致したとみなされなかったことになる．この文字は縦軸側に変更する際に，新たに挿入する必要がある．

さらに，一度に対角線方向にたどった場合は，その元の位置の縦軸側と横軸側の両方ともが一致しなかったわけであるから，縦軸側の文字を上書きしてよいことになる．対角線上のこの処理を上方向と左方向の2段階に分ければ，これは削

リスト4

```
# lcs を作成する
function LCS(a, b,  i, j, r) {
    for(i = 1; i <= jlength(a); i++) {
        for(j = 1; j <= jlength(b); j++) {
            if(jsubstr(a, i, 1) == jsubstr(b, j, 1)) {
                lcs[i, j] = lcs[i-1, j-1]+1
            }
            else if(lcs[i, j-1]+0 > lcs[i-1, j]+0) { lcs[i, j] = lcs[i, j-1]+0 }
            else { lcs[i, j] = lcs[i-1, j]+0 }
        }
    }
# lcs から最長共通部分列を生成
    r=""; i--; j--
    while(i > 0 || j > 0) {
        if(i > 0 && j > 0 && lcs[i-1, j-1]+0 == lcs[i, j]) {i--; j--}
        else if(j <= 0 || i > 0 && lcs[i-1, j]+0 == lcs[i, j]) { i-- }
        else if(i <= 0 || j > 0 && lcs[i, j-1]+0 == lcs[i, j]) { j-- }
        else { r = jsubstr(a, i, 1) r ; i-- ; j-- }
    }
    return r
}
function LCSinit() { return ""}
```

リスト5

```
function lclines(scr1, scr2,  i, j, r) {
# lcs を作成する
    for(i = 1; i <= scr1["LINES"]; i++) {
        for(j = 1; j <= scr2["LINES"]; j++) {
            if(redrawline(scr1, i) == redrawline(scr2, j)) {
                lcs[i, j] = lcs[i-1, j-1]+1
            }
            else if(lcs[i, j-1]+0 > lcs[i-1, j]+0) { lcs[i, j] = lcs[i, j-1]+0 }
            else { lcs[i, j] = lcs[i-1, j]+0 }
        }
    }
# lcs から書き換え指令を生成
    r=""; i--; j--
    while(i > 0 || j > 0) {
        if(i > 0 && j > 0 && lcs[i-1, j-1]+0 == lcs[i, j]) {
            r = "行␣" j "␣を行␣" i "␣に␣上書き\n" r
            j--; i--
        }
        else if(j <= 0 || i > 0 && lcs[i-1, j]+0 == lcs[i, j]) {
            r = "行␣" i "␣を␣削除\n" r
            i--
        }
        else if(i <= 0 || j > 0 && lcs[i, j-1]+0 == lcs[i, j]) {
            r = "行␣" j "␣を行␣" i "␣の後に␣挿入\n" r
            j--
        }
        else { i-- ; j-- }
    }
    return r
}
```

除と挿入を続けて行なうことになる．実際，削除と挿入は上書きと原理的には等価であるから，一種の操作の最適化を行なったことになる．これが，前節での対角動作の優先の理由であった．

最後に，角にいて次の角を探すのを始めるのは，その位置の横軸側の文字を縦軸側に利用するのであるから，何も特別な操作をする必要はない．

これをプログラムにすると**リスト5**のようになる．ここでは，先に説明したように最終目的のために行を単位とするような列について最長部分列の長さを計算することにしてある．スクリーンの行をもってくるredrawline( )を通して，前述のライブラリとリンクしている．書き換え指令は，とりあえず分かりやすいように文章で表示するようにした．

ここまでくればあと一息であるが，完成したわけではない．ここで思い出されるのが，行番号を指定するエディタでの編集作業の経験である．通常編集作業は前のほうから行なうが，作業の進捗によってテキストの増減が起きる．これは後ろのほうに波及して，下書きにメモした編集用のテキストとは位置が大きく変わってしまうことがよくある．これを避けるために，下書き付きの編集では，後ろのほうの変更から先に片付けるというノウハウが伝授されている．これと同じことが，この場合もある．

たとえば，“行1を削除，行2を削除”を忠実にこの順で行なうと，実際には行1と行3が削除されてしまう．このために，前の何番目の行が今の何番目の行に対応するのかを記録した表を持っている，という方法も考えられる．しかし，これは面倒である．先のノウハウを利用したほうが簡単であろう．つまり，挿入や削除は行番号の大きいほうから行なえばよい．

もう1つ問題がある．これは，挿入があると一番下の行が消えてしまうという物理的な画面の制約である．これを防ぐには挿入よりも削除を先に行なうことである．また，このことにより，今度は挿入の実行順序を行番号の小さいほうからにし，挿入する場所は変更後の位置を示すのがよい．上書きも挿入削除の後に行なうのならば，変更後の位置に行なってかまわない．

このように考えてプログラムをすると，**リスト6**のようになる．画面操作は上書き（ovw）挿入（ins）削除（del）の3つに分けて格納する．ここで，再利用する場合にどの行を再利用するかをnoeという文字列に格納しているが，これは実は単なるフラグとして使っているだけである．利用目的はちょっとした最適化で，削除を先に行なうと一番下の行に空行が挿入されることを利用しようというものである．つまり，下にできる空行をわざわざ挿入で消し去る必要はなく，単に上書きすれば十分ということである．これができるためには，その挿入操作をする行より後に削除や上書きや再利用がないことが条件となる．これを確認する目的でnoeを使っている（デバッグにも使えるが）．また，同じように，挿入でいずれは消えてしまうはずの一番下の行をわざわざ削除する必要はないので，この場合は再利用フラグを立てて将来の挿入に任せるようにしている．

あと必要なのは，上書きにおける行末までのクリアの問題であるが，これは練習問題としておこう．

次回以降の予告

画面書き換えの最適化ができたところで，さて本題の画面エディタの作成に入るのが本筋であるが，筆者の準備の都合で多少時間をいただきたい．次回は話題をガラリと変えて，データベースとCAIの真似事を取り上げてみたいと考えている．

参考文献
- 『岩波講座ソフトウェア科学3 アルゴリズムとデータ構造』 石畑清，岩波書店，1989
- 『The string-to-string correction problem』 R. Wagner and M. Fischer, J. ACM 21, 1, 1974

リスト6

```
function lclines(scr1, scr2,  i, j, ovw, del, ins, noe) {
    for(i = 1; i <= scr1["LINES"]; i++) {
        for(j = 1; j <= scr2["LINES"]; j++) {
            if(redrawline(scr1, i) == redrawline(scr2, j)) {
                lcs[i, j] = lcs[i-1, j-1]+1
            }
            else if(lcs[i, j-1]+0 > lcs[i-1, j]+0) { lcs[i, j] = lcs[i, j-1]+0 }
            else { lcs[i, j] = lcs[i-1, j]+0 }
        }
    }
    i--; j--
    ovw = del = ins = noe = ""
    while(i > 0 || j > 0) {
        if(i > 0 && j > 0 && lcs[i-1, j-1]+0 == lcs[i, j]) {
            ovw = "OL␣" j "\n" ovw; j--; i--
        }
        else if(j <= 0 || i > 0 && lcs[i-1, j]+0 == lcs[i, j]) {
            if(ovw ins noe != "") { del = del "DL␣" i "\n" }
            else { noe = noe i "\n" }
            i--
        }
        else if(i <= 0 || j > 0 && lcs[i, j-1]+0 == lcs[i, j]) {
            if(ovw del noe == "") { ins = "OL␣" j "\n" ins}
            else { ins = "IL␣" j "\n" ins }
            j--
        }
        else { noe = noe i "\n"; i-- ; j-- }
    }
    return del ins ovw
}
```

ウィンドウシステムのプログラミングパラダイム

MACINTOSHからX WINDOW,さらにSTARまで ウィンドウシステム上のプログラミングの本質を衝く

中川幸男

ウィンドウシステムのプログラムは，どこが面倒なのか？
Aboutダイアログの組み込みと通信プロトコルを例に考えてみよう

Event Handling

前回はウィンドウを作って，その中に文字列を表示する場合を考えてみたが，今回はウィンドウシステムでの入力処理に欠かせない「イベント」の取り扱いについてお話ししよう．

どんなシステムでも，必ず外部からの入力を受け取り，それに従って処理を行なっているが，この入力の扱いはキャラクタベースのシステムとウィンドウシステムとではかなり異なっている．なぜだろうか？　それは入力として対処すべき対象がウィンドウシステムでは必然的に種類が多いことに起因している．キャラクタベースのシステムでは，外部からの入力の大部分がキーボードからであるのに対し，ウィンドウシステムではキーボードに加えてマウスなどのポインティングデバイスからの入力にも対処しなくてはならない．また，キャラクタベースの場合には通常出力用のデバイス(つまりディスプレイ)は1つしかないのに対し，ウィンドウベースの場合にはマルチウィンドウといって複数のウィンドウを同時に画面上に表示できるのが普通である．

マルチウィンドウのシステムでは，ちょうど机の上に何枚かの書類を広げて仕事をしているときのようにいくつかのウィンドウが画面上に表示されることになる．このとき机の上の書類で一番上になっているものにしか書き込みができないのと同様に，ウィンドウシステムでもウィンドウのどれか1つだけが入力を受け付けるようになっている．この入力可能になっているウィンドウのことを「インプットフォーカス」とか「アクティブウィンドウ」などと呼ぶが，通常これは画面上の一番手前になる．当然，これ以外のウィンドウに入力したければインプットフォーカスを移す必要がある．つまり下のほうに隠れている書類を一番上(手前)に持ってくるのと同じように，下のほうに埋もれているウィンドウを手前に表示し直す必要があるわけだ．手前にくるウィンドウの一部が他のウィンドウに覆われていて見えていなかったりすると，その部分を書き直す必要がでてくる．

これはどういうことかというと，液晶やCRTなどのディスプレイに絵や文字を表示させるために，通常は「ディスプレイバッファ」というメモリ領域にビットマップ情報を書き込んでいる．このメモリ領域は物理的な画面の大きさの分しかなく，すべてのウィンドウで共有されている．そのため，重なりあっている部分は常に一番上になっているウィンドウによって「上書き」されることになる．つまり下になって隠れてしまった領域の情報は，ディスプレイから失われてしまうのである．

このため，通常はディスプレイバッファとは別の領域(オフスクリーンビットマップなどと呼ぶ)を各ウィンドウごとに用意し，一度そこに書いてからそれをディスプレイ領域に転送する．これはビットマップディスプレイのシステムではある程度常識的に行なわれている．この転送動作のことをBitBltなどと呼ぶのを聞いたことがあると思う．

だから，ディスプレイ上で重なりあって隠れていた部分が最上部になる時点で，この隠れていた部分を転送して書き直す作業が必要なのである．この書き直しをするタイミングは「自分が最上位になった」

という「状態の変化」から判断しているわけだ．したがって，マウスやキーボードが押されたという直接的な事象から，画面の書き直し要求といった間接的な事象が起こる．その結果こういった「状態の変化」も１つの入力として扱う必要があるということになる．このようにウィンドウシステムになっただけでキャラクタベースのシステムに比べて，対処すべき要因がぐっと増える．

さて，こうした外部からの「事象」や「対処すべき要因」のことを一般に「イベント」と呼ぶ．通常はアイドル状態で何らかのイベントが発生するのを待っており（これをイベントループと呼ぶ，詳細は後述），イベントが発生した時点でその種類に応じた処理をするシステムのことを「イベントドリブン」または「イベント駆動型」のシステムと呼ぶ．現在使われているウィンドウシステムは，そのほとんどがイベントドリブンであるといっても過言ではない．したがって，ウィンドウシステムを理解するためにはまずイベントの処理とはどういうものかを理解する必要があるだろう．

Event Types

ウィンドウシステムでは，それぞれあらかじめ扱うイベントの種類が定められているのが普通だ．つまりそのウィンドウシステムを使ううえで発生するであろうイベントの種類が，あらかじめ限定されているということになる．前回も取りあげたMacでは，**表1**に示した種類のイベントが定義されている．

これがX Windowになるとまた違った定義をしている．**表2**にその定義を示す．詳細については他の文献を参考にしていただくとして，それぞれのウィンドウシステムごとにどういった思想でイベントを扱っているかを，この定義の違いから感じとっていただければ十分だと思う．

次にイベントがシステムの内部でどのように扱われているのかを見てみよう．以下にMacの場合を示す．

```
typedef struct EventRecord {
        int         what;
        long        message;
        long        when;
        Point       where;
        int         modifiers;
}       EventRecord;
/* THINK C のEventMgr.hより抜粋 */
```

表1 Macのイベントコード

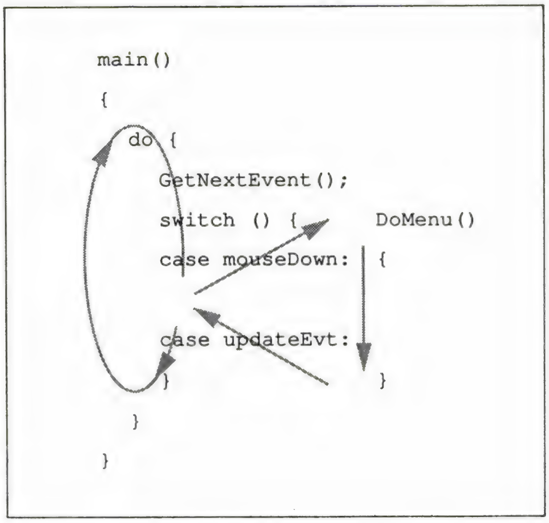
```
enum {
        nullEvent,        /* = 0 */
        mouseDown,        /* = 1 */
        mouseUp,          /* = 2 */
        keyDown,          /* = 3 */
        keyUp,            /* = 4 */
        autoKey,          /* = 5 */
        updateEvt,        /* = 6 */
        diskEvt,          /* = 7 */
        activateEvt,      /* = 8 */
        /* event 9 is no longer used*/
        networkEvt = 10,
        driverEvt,        /* = 11 */
        app1Evt,          /* = 12 */
        app2Evt,          /* = 13 */
        app3Evt,          /* = 14 */
        app4Evt           /* = 15 */
};      /* THINK CのEventMgr.hより抜粋 */
```

それぞれの意味は名称を見ていただけばだいたいの見当はつくと思う．もっとも上の15種類のうち，app1Evtからapp4Evtまでのイベントはアプリケーションごとに定義してもよいことになっており，通常はあまり使わないので実際には残りの11種類のイベントだけを相手にすればよいということになる．また，厳密にいうとnullEvent，diskEvt，networkEvt，およびdriverEvtの４つはウィンドウとは直接関係がない．したがって，マウス関係，キーボード関係，およびウィンドウの状態変化の３種類がMacでのウィンドウ関係の代表的なイベントというわけだ．

表2 X Windowのイベントコード

```
#define KeyPress 2
#define KeyRelease 3
#define ButtonPress 4
#define ButtonRelease 5
#define MotionNotify 6
#define EnterNotify 7
#define LeaveNotify 8
#define FocusIn 9
#define FocusOut 10
#define KeymapNotify 11
#define Expose 12
#define GraphicsExpose 13
#define NoExpose 14
#define VisibilityNotify 15
#define CreateNotify 16
#define DestroyNotify 17
#define UnmapNotify 18
#define MapNotify 19
#define MapRequest 20
#define ReparentNotify 21
#define ConfigureNotify 22
#define ConfigureRequest 23
#define GravityNotify 24
#define ResizeRequest 25
#define CirculateNotify 26
#define CirculateRequest 27
#define PropertyNotify 28
#define SelectionClear 29
#define SelectionRequest 30
#define SelectionNotify 31
#define ColormapNotify 32
#define ClientMessage 33
#define MappingNotify 34
#define LASTEvent 35
/* must be bigger than any event # */
```

Macよりイベントの種類がだいぶ多いのは，ウィンドウのサイズ変更などの画面の再表示にかかわるイベントがすべて独立しているからである．そのほか，カラーマップの変更などのイベントが追加されている．



リスト１のプログラムは，１つのswitch文を無限ループして実行するようになっている．caseによっては下位のswitch文が呼ばれるが，そこではループはしない．

「いつ」「どこで」「どんな」イベントが発生したかを表わす構造体として定義されているわけだ．またmessageはイベントの種類（what）によって意味が変わってくる．たとえばキーボード関係のイベント(keyDown，keyUp，autoKey)では押されたキーに対応したキーコードやキャラクタコードなどが32bitの中に収められているといった具合だ．したがって適当なマスクビットでANDを取ることで目的のコードを得ることができる．また，modifiersはイベントが発生したときの修飾キー(optionキーやcommandキーなど)の状態などが格納されている．

システム内外で発生したイベントは，ハードウェア割り込みなどでまずOSに渡される．次にOSはこれをEvent Recordの形にして，イベントキュー(待ち行列)に登録する．ここまでがOSのEvent Managerの仕事だ．イベントの種類によっては例外的にイベントキューには入れられないものもあるが，とりあえずはキューに入れることを前提に説明する．一方ユーザーアプリケーションはToolboxのEventManagerを使って，このキューから１個ずつイベントを取り出しては逐次適切な処理を行なうという動作を繰り返している．この繰り返し動作のことを「イベントループ」と呼ぶ．

イベントループを組む理由の１つは，イベントがシステムの都合とは関係なくランダムかつ非同期に発生するためである．つまりいつ，どんなイベントが発生するかあらかじめ予測できないため，できるだけ短い周期でイベントキューを監視してイベントが発生したら速やかにこれに対処する必要があるからである．もしこれを怠ると，ユーザーがマウスをクリックしてもシステムがなかなか反応しない，ユーザーにとって非常に使い勝手の悪いシステムになってしまう．

GetNextEvent()

それでは実際にどのようにしてイベントを扱うのかを見ていこう．**リスト1**にごく一般的な例を示した．ここで「ごく一般的」と断ったのは普通に書くとこうするだろうという意味で，もっと別の書き方もあるということを言いたかったのだ．「別の書き方」はもう少し後のほうで紹介するとして，とりあえずは一般的な方法を説明しておこう．このプログラム自体はイベントだけではなくてMacのプログラムの基本を示すためのものだが，イベントなどもかなりまともに扱っているので参考になるだろう．

Macでイベントを「拾う」ためには，通常GetNextEvent()というToolboxの関数を使う．ここで第１引数はイベントマスクと呼ばれ，16ビットのビット単位で受けたいイベントの種類を指定する．もしすべてのイベントを拾いたいのなら，everyEventを指定すればよい．

第２引数には指定したイベントが発生したときにそれを格納するためのEventRecord型変数へのポインタを渡す．また，戻り値はBOOLEAN型で第１引数で指定したイベントがあった場合にはTRUEが，さもなければFALSEが返される．なお，GetNextEvent()はイベントが発生するまで待つのではなく，呼ばれた時点でのイベントキューの内容を調べ，イベントの有無にかかわらず即座にリターンする関数である．

イベントキューは，通常の待ち行列と同様にFIFO(First In First Out)であるため，通常はイベントの発生した順番通りにキューに積まれその順番通りに取り出されることになる．しかし，一部の特殊なイベント(activateEvtなど)はイベントキューを経由しないため，その他のイベントよりもプライオリティが高い扱いとなっている．さらに第１引数で，ある特定のイベントしか取らないように指定した場合にも，その指定に該当しないイベントはたとえキューの中にあっても読み飛ばされるようになっている．したがって，必ずしもイベントの発生した順番通りに受け取れるとは限らないので注意が必要である．

また，GetNextEvent()を実行した時点でキューが空か，もしくは指定したイベントがなかった場合には第２引数のEventRecordにはnullEventが格納されFALSEが返される．FALSEが返されたということは「処理すべきイベントがなかった」ということなので，通常はまたすぐにGetNextEvent()を実行することになる．ここで気を付けなければいけないのは，イベントループを実行している間はCPUは基本的にそのアプリケーションに占有されているということだ．そこで適当な間隔でSystemTask()という関数を呼ぶ必要がある．つまりシステムを始めとする，ほかのプログラムに一時的にCPUを返してあげなくてはならない．そうしないとシステムとして必要な処理(たとえ

```
void
DoMenu(mResult)
long mResult;
{
	int theItem, theMenu;
	char name[256];
	int temp;

	theMenu = HiWord(mResult);
	theItem = LoWord(mResult);

	switch (theMenu) {
		case appleID:		/* apple menu */
			if (theItem == aboutCommand)
				mac_aboutcmd();
			else {
				GetItem(myMenus[appleM], theItem, name);
				temp = OpenDeskAcc(name);
				SetPort(myWindow);
			}
			break;
		case fileID:		/* file menu */
			pleaseStop = TRUE;
			break;
		case editID:		/* edit menu */
			/* edit function omitted */
			break;
	}
	HiliteMenu(0);
}

main()
{
	InitGraf(&thePort);
	InitFonts();
	FlushEvents(everyEvent, 0);
	InitWindows();
	InitMenus();		/* init menu from 'DRVR' resource */
	TEInit();	/* init Text Edit routine */
	InitDialogs(0L);
	InitCursor();

	SetRect(&dragRect, 4, 24,
			screenBits.bounds.right-4,
			screenBits.bounds.bottom-4);
	pleaseStop = FALSE;
	myWindow = GetNewWindow(windowID, &wRecord, -1L);
	SetPort(myWindow);
	txRect = thePort->portRect;
	InsetRect(&txRect, 4, 0);
	textH = TENew(&txRect, &txRect);

	do {					/* main event loop */
		SystemTask();	/* dispatch system task */
		TEIdle(textH);
		if (GetNextEvent(everyEvent, &myEvent)) {
			switch (myEvent.what) {
				case mouseDown:		/* click */
				switch (FindWindow(myEvent.where, &whichWindow)) {
					case inSysWindow:
						SystemClick(&myEvent, whichWindow);
						break;
					case inMenuBar:
						DoMenu(MenuSelect(myEvent.where));
						break;
					case inDrag:
						DragWindow(whichWindow, myEvent.where, &dragRect);
						break;
					case inContent:
						if (whichWindow != FrontWindow())
							SelectWindow(whichWindow);
						else {	/* move insert point */
							GlobalToLocal(&myEvent.where);
							extended = (Boolean)BitAnd(myEvent.modifiers,
										shiftKey);
							TEClick(myEvent.where, extended, textH);
						}
						break;
				}
				break;
				case keyDown:	/* key in */
				case autoKey:
					theChar = (char)BitAnd(myEvent.message, charCodeMask);
					if (BitAnd(myEvent.modifiers, cmdKey) != 0)
						DoMenu(MenuKey(theChar));
					else
						TEKey(theChar, textH);
					break;
				case activateEvt:
					if (BitAnd(myEvent.modifiers, activeFlag) != 0) {
						TEActivate(textH);
						DisableItem(myMenus[editM], undoCommand);
					} else {
						TEDeactivate(textH);
						EnableItem(myMenus[editM], undoCommand);
					}
					break;
				case updateEvt:		/* redraw */
					BeginUpdate((WindowPtr)(myEvent.message));
					EraseRect(&thePort->portRect);
					TEUpdate(&thePort->portRect, textH);
					EndUpdate((WindowPtr)(myEvent.message));
					break;
				default:
					break;
			}
		}
	} while (!pleaseStop);
}
```

リスト1　このプログラムはInside MacintoshのVolume1に出ているサンプルプログラムを，筆者が一部手直しをしてC言語で書き直したものだ．もともとはPascalで書かれている．

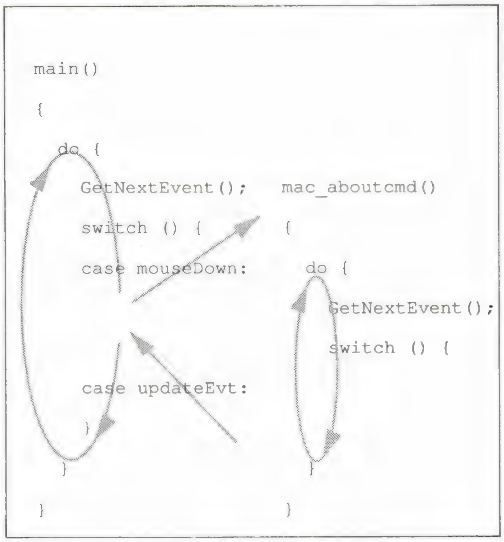
ば時計のアップデートなど)ができなくなってしまう．

GetNextEvent( )でTRUEが返された場合には，myEvent.whatを見るとどんなイベントが発生したのか知ることができる．イベントの種類はあらかじめ限定されているのでswitch文を使ってイベントに応じた処理をするのが普通のスタイルだ．なお，ここではべた書きでそれぞれのイベントに対応した処理を書いてあるが，イベントドリブンのシステムではやたらとswitch文が出てくる傾向があるので，できるだけ関数に分割したほうがベターだろう．

細かいプログラムのスタイルはともかくとして，具体的にイベントの扱いを考えてみる．試しにmouseDownの処理を見てみよう．単純にマウスがクリックされたということだけを検出すればよいという場合はめったになく，普通はどこでクリックされたのかを調べ，それに応じて処理も変える必要がある．このためにFindWindow( )という関数が用意されている．これはWindowManagerの関数で，マウスが押された場所を画面左上を原点(0,0)とした画面上の座標(Point型)を引数として，以下のような整数値を返す．

```
void
mac_aboutcmd()
{
        GrafPtr eventport;
        Rect r;
        EventRecord ev;
        WindowPtr w;
        boolean flbitmap;

        InitCursor();
        SetRect(&r, 0, 0, 340, 120);
        pushport(w = NewWindow(nil, &r, "\p", false, altDBoxProc, -1L, false, 0));
        positionaboutwindow((DialogPtr) w, screenBits.bounds);
        zoomport(w, true);
        drawabout(w);           /* draw about dialogue */
        while(true){            /* another event loop */
            if(!GetNextEvent(everyEvent, &ev))
                continue;
            SystemTask();
            switch(ev.what) {
                case keyDown:
                case autoKey:
                case mouseDown:
                    goto exit;          /* return to main menu */
                case updateEvt:     /* screen savers may cause this */
                    if((WindowPtr)(ev.message) == w) {
                        pushport(w);
                        BeginUpdate(w);
                        drawabout(w);           /* redraw about dialogue */
                        EndUpdate(w);
                        popport();
                    }
                    break;
                default:
                    break;
            }
        }
    exit:
        HideWindow(w);
        zoomport(w, false);
        DisposeWindow(w);
        popport();
        FlushEvents(mDownMask + keyDownMask, 0);
}
```



リスト２の関数は，メインのルーチンと同じようにswitch文を無限ループして実行している．メインのイベント処理ルーチンから呼ばれているにもかかわらず，独自のイベント処理ルーチンを持つのである．しかし，ここからはほとんどのイベントで脱出してしまう．

リスト2　実際に動いているところをお見せできないのが残念だが，実はこのアバウトダイアログはちょっと凝っていてズームアップするような感じでウィンドウが開くのだ．実はこれは先日某ネットからソースコード込みでダウンロードしてきたフリーウェア(Mac版のKA9Q)のプログラムの一部を借用したもの．

```
/* FindWindow values */
enum {
        inDesk,       /* = 0 */
        inMenuBar,    /* = 1 */
        inSysWindow,  /* = 2 */
        inContent,    /* = 3 */
        inDrag,       /* = 4 */
        inGrow,       /* = 5 */
        inGoAway,     /* = 6 */
        inZoomIn,     /* = 7 */
        inZoomOut,    /* = 8 */
};
/* THINK CのWindowMgr.hより抜粋 */
```

また，このFindWindow( )の第２引数にはWindowPtr型の変数(whichWindow)を渡すが，これは複数のウィンドウが同時に表示されている場合に，どのウィンドウ内でマウスがクリックされたかを示すために使われる．ともあれ，このリターン値をswitch文で判断し，それぞれの場合に応じた処理をすることになる．さらにFindWindow( )からinMenuBarが返された場合には，今度はメニューバーの中のどのメニューで，何番目のアイテムが選ばれたのかを調べなくてはならない．そのために，MenuManagerの中のMenuSelect( )という専用の関数を使って調べている．

MenuManagerの話が出たので，少しメニューについて説明しておこう．代表的なメニューにはある定まった場所でマウスなどをプレス(クリックしたままボタンを離さない状態のこと)することによって表示される「プルダウン」方式と，画面上のどこにでも表示される「ポップアップ」方式とがある．Macの場合には(通常は)前者の方式で，画面の一番上に「メニューバー」と呼ばれる白い帯状のものが表示されているのが普通だ．また，Macの場合にはApple社が出しているガイドラインを守って作ればどのアプリケーションも似たようなメニューになるように考えられている．たとえば一番左端がアップルメニュー，次がFileメニュー，その次がEditメニューといった具合．逆にいえば各プログラマがそのようにプログラムを書く必要があるわけだ．実際のところ，前回の例のようにプログラムがさぼってしまえばまったく何もないメニューというのもできてしまうわけで，このへんはプログラムが何もしなければ最低限のメニューは表示してくれるぐらいの配慮がToolboxにあったら... と思ってしまう．

これでウィンドウシステムでの入力は完璧か？　というとそうでもない．メニューはあくまでもあらかじめ選択肢がある程度決まっているときには有効だが，新

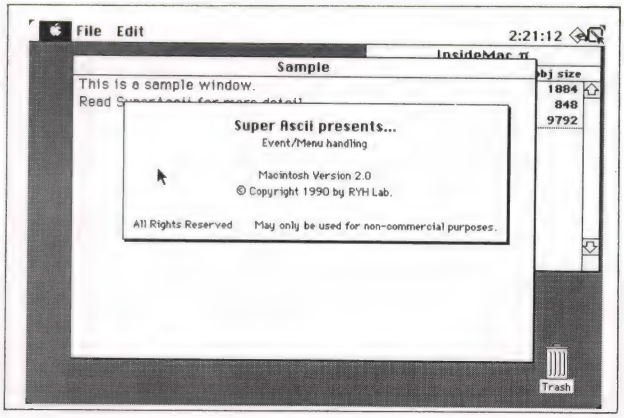


図1 Aboutダイアログが表示された画面
画面中央のウィンドウがアバウトダイアログである．ダイアログといっても大した会話ができるわけではない．表示をやめて，メインルーチンに戻る以外のことは何もしない．

たにファイルをストアする場合などは，どうしてもファイル名をユーザーに入力してもらう必要がある．このような場合には基本的にメニューは使えない．ではどうするか？　このような場合に使うのがMacでいうところの「ダイアログ」である．ダイアログというのは一種のウィンドウだが，メインのウィンドウに対してファイル名の入力などといった補助的な目的に使う特殊なウィンドウと考えればよいだろう．

また，Macの場合にはメニューの項目の中に「Open...」のようにコマンド名の後ろにエリプス(ピリオドが3つ並んだような記号)というのが付いているものがある．こういうコマンドを選択すると，たいていダイアログが表示されると思っていて間違いない．今回は比較的単純な例で，特にファイルを扱ったりはしていないので，アップルメニューの中にAbout...というコマンドを追加してみた．これもApple社のガイドラインに示されていることで，各アプリケーションはそのプログラムの簡単な説明をこの「アバウトダイアログ」によって表示できるように作ることが推奨されている．これを実行すると**図1**のようなダイアログが表示される．

Event Loop

ここでイベントループの基本を整理してみよう．1)イベントはシステムやプログラムの都合とは無関係に，非同期に発生する．2)システム(OS)はこれをFIFOのイベントキューに入れる．3)アプリケーションはキューから1つずつイベントを取り出して順番に処理する．このとき，さまざまな種類のイベントが発生するので，その種類に応じて処理を変える必要がある．4)アプリケーションがイベントキューを監視するタイミングが遅い(ループを回る周期が長い)とキューにどんどんイベントがたまってしまう．5)逆にイベントが発生する間隔が長いと，アプリケーションはイベント待ちのループをむなしく回り続けなければならない(Active Wait)．このため，システムによっては(MacのSystemTask()のような)特定の処理をループの中で呼ぶことを義務付けているものもある．6)イベントは極力1カ所で受ける．

以上が，最も基本的なイベントループの作り方だ．もともとMacの場合には，Apple社からのガイドラインによって，「モードレス」のプログラムを作るように推奨されているし，システム自体がシングルタスク(注: MultiFinder も基本的にはシングルタスクで，ユーザーが明示的にアプリケーションを切り替えることにより擬似的にマルチタスクを実現しているにすぎない)であるために，どんなプログラムでもこんな書き方にできるように思える．しかしアプリケーションによっては，単純なイベントループでは対処しきれない場合も出てくる．

State Matrix

さて，ここで1つ考えていただきたいことがある．**リスト1**のイベントループでは一応すべてのイベントを拾うようになっている．したがって，できるだけ短い周期でここに戻ってくれば，一応のイベントには対処することができるはずである．ところが，世の中そんなに甘くはない．たとえば先ほどの例のようにアバウトダイアログを表示させたとしよう．このダイアログは何かキーが押されるか，マウスがクリックされるまで画面から消えない．つまりここでダイアログを消すタイミングを把むためにメインのイベントループとは独立に別のイベントループを作っているのだ(**リスト2**参照)．

したがって，ダイアログを表示している間は，本来のプログラムの流れとは別の「モード」に入っていることになる．**リスト1**のイベントループでcaseのそれぞれにモードの条件判断を追加すれば，形式的には1つのイベントループにまとめることもできるが，モードはなくならない．また，プログラムの可読性も下がる．このようにユーザー(人間)からのインタラクションだけを相手にしているごく普通のプログラムでは，モードの数も少なくモードごとのイベントループも簡単にできるから，これでも特に問題はない．ところが，相手が計算機である通信関係のプログラムの場合には，かなり問題が出てくる．

通信関係のプログラムでは，相手とのデータのやり取りによって，非常に多くの状態(ステート)を管理しなければならないのが普通だ．このとき，相手からのデータをイベントとして捉えることができるが，イベントの種類とそれに対応するステートの種類が非常に多いためにswitch文やif文では見通しの悪いプログラムとなってしまい，普通の神経の持ち主にはデバッグやメンテナンスはできなくなってしまう．そこで登場するのが「イベント-ステートマトリックス(単にステートマトリックスなどと呼ぶ)」である．

リスト3にAX.25という通信プロトコルのステートマトリックスをプログラムに

落とした例を示す．ステートマトリックスとは，たとえばS1(ディスコネクト)の状態のときにSABM(接続要求)というイベントを受け取ったらUA(接続了解)を送り，内部状態をS5(データ転送中)に移すというようにイベントに応じた処理と状態の変化をテーブル(マトリックス)にしたものだ．そして，このテーブルをそのままジャンプテーブルにしたのが**リスト3**である．この考え方は，当然ウィンドウでのイベント処理にも応用できる．**リスト1，2**の例なら2行のマトリックスになる．このような手法を使えばプログラムが見やすくなるばかりでなく，いちいち判断をせずに直接目的の処理にジャンプできるので，処理時間も短くできるというメリットがある．

さて，これを**リスト1**のメインイベントループの中に埋め込んだ場合を考えてほしい．相手からの要求に対し，即座に処理を行なって返事を返す必要があるわけだが，ここに先ほどのアバウトダイアログの処理が入ったらどうなるだろう？

アバウトダイアログのイベントループの中で通信関係のイベントをちゃんと拾って処理をしない限り，相手からのイベントはキューにたまったままとなり，最悪の場合は通信が途絶えてしまうという悲惨な結果になるのである．またステートマトリックスを使おうとしても，ステートが2次元になるためマトリックスが結果として3次元になってしまう．

それでは，このような場合に対処するにはどうしたらよいか？　というお話は次回のお楽しみということにしよう．次回はシングルタスクのMacから少し離れて，マルチタスクのシステムでのウィンドウシステムの話をする予定である．どうなることやら．

今回の環境

Hardware	Macintosh Plus, Apple Computer
OS	漢字 Talk 6.0.4, Apple Computer
Compiler	THINK C 3.02, Symantec

```
/* Event definition */
#define RR          0
#define REJ         1
#define RNR         2
#define UA          3
#define SABM        4
#define MaxEvents   5

/* State definition */
#define Disconnected     0      /* S1 */
#define LinkSetup        1      /* S2 */
#define FrameReject      2      /* S3 */
#define DisconnReq       3      /* S4 */
#define InfoXfer         4      /* S5 */
#define RejectSent       5      /* S6 */
#define WaitAck          6      /* S7 */
#define DeviceBusy       7      /* S8 */
#define RemoteBusy       8      /* S9 */
#define BothBusy         9      /* S10 */
#define WaitAckDevBusy 10       /* S11 */
#define WaitAckRemBusy 11       /* S12 */
#define WaitAckBothBusy 12      /* S13 */
#define RejSentDevBusy 13       /* S14 */
#define RejSentRemBusy 14       /* S15 */
#define RejSentBothBusy 15      /* S16 */
#define MaxStates        16

typedef enum {
        S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15, S16
} States;

typedef struct {
        int         nextState;
        int         (*action)();
} TableElement;

typedef TableElement        Table[MaxEvents];

extern int      Noop();
extern int      SendUA();
extern int      SendDM();
extern int      SendData();
extern int      SendSABM();

Table   matrix[MaxStates] = {     /* state matrix */
{{S1, Noop}, {S1, Noop}, {S1, Noop}, {S1, Noop}, {S5, SendUA}},
{{S2, Noop}, {S2, Noop}, {S2, Noop}, {S5, Noop}, {S5, SendUA}},
{{S3, Noop}, {S3, Noop}, {S3, Noop}, {S3, Noop}, {S5, SendUA}},
{{S4, Noop}, {S4, Noop}, {S4, Noop}, {S1, Noop}, {S1, SendDM}},
{{S5, SendData}, {S5, SendData}, {S9, Noop}, {S5, Noop}, {S5, SendUA}},
{{S6, SendData}, {S6, SendData}, {S15, Noop}, {S2, SendSABM}, {S5, SendUA}},
{{S7, SendData}, {S7, SendData}, {S12, Noop}, {S2, SendSABM}, {S5, SendUA}},
{{S8, SendData}, {S8, SendData}, {S10, Noop}, {S2, SendSABM}, {S8, SendUA}},
{{S5, SendData}, {S5, SendData}, {S9, Noop}, {S2, SendSABM}, {S5, SendUA}},
{{S8, SendData}, {S8, SendData}, {S10, Noop}, {S2, SendSABM}, {S8, SendUA}},
{{S11, SendData}, {S11, SendData}, {S13, Noop}, {S2, SendSABM}, {S8, SendUA}},
{{S7, SendData}, {S7, SendData}, {S12, Noop}, {S2, SendSABM}, {S5, SendUA}},
{{S11, SendData}, {S11, SendData}, {S13, Noop}, {S2, SendSABM}, {S8, SendUA}},
{{S14, SendData}, {S14, SendData}, {S16, Noop}, {S2, SendSABM}, {S8, SendUA}},
{{S6, SendData}, {S6, SendData}, {S15, Noop}, {S2, SendSABM}, {S5, SendUA}},
{{S14, SendData}, {S14, SendData}, {S16, Noop}, {S2, SendSABM}, {S5, SendUA}}
};

int
StateMachine(event, state)
int event, state;
{
        (*matrix[state][event].action)(event);
        return (matrix[state][event].nextState);
}
main()
{
        int state = S0;
        while(1) state = stateMachine(getEvent(), state);
}
```

リスト3　AX.25はアマチュア無線を使ったパケット通信で使われているプロトコルである．ちなみにRRはRecieve Ready，REJはREJect，RNRはReceive Not Readyのそれぞれ略である．

EP news

Electronic Publishing

このEPnewsは，EP(Electronic Publishing) に関する情報のページである．

従来，Electronic Publishingというのは，CD-ROMなどの電子化された媒体による電子出版物およびいわゆるDTP (DeskTop Publishing)のような電子化された編集／印刷プロセスを意味していた．現状でもこの認識に誤りはない．

本来，出版 (Publishing) とは，情報を広く伝達することを意味している．その伝達媒体が圧倒的に“紙”に印刷された文字だったわけである．しかし，今，情報は文字や静止画で表現されるものだけではなく，サウンド，動画など多岐にわたってきている．これらを伝達するには，あらゆるメディアにまたがったテクノロジーを統合していかなければならない．このことは，出版／印刷分野とコンピュータとの統合により実現できる．つまり，EPは統合の接点という位置付けがされ，EPに関する情報は，とりもなおさずコンピュータに関する総合的情報になる．このような視点で，このコーナーを読んでいただきたいと考える．

今号は，EP講座には，異機種間，ネットワークでの文書ファイルの交換に関するテクノロジーを取り上げた．EPキーワードには，前号に引き続きフォントを取り上げている．トピックスには，米国で開催された“Digital World”のレポート，業界動向には，幕張メッセで開催された“伸びゆく軽印刷展”から見た最近の傾向を紹介している．

基礎講座

文書交換のための共通文書フォーマットの動向

文書の共有化，効率的文書交換をめざして

テクニカルアナリスト 村上 裕一

はじめに

近年のオフィスにおける文書作成環境の向上には目覚ましいものがある．オフィスのほとんどの文書は「タイプ化」されているのが常識的となり,「手書き資料」を目にすることはまれである．このことは，各種ワードプロセッサ（以下ワープロ）のオフィスへの浸透が大きな要因となっており，日本における文書作成環境が手書きの清書からワープロへと変貌したといっても過言ではないであろう．

しかしながら，オフィスに導入されているワープロ機種の多様化により，それぞれのワープロにおいて作成される文書フォーマットの互換性が損なわれ,「文書の共有化」や「効率的な文書交換」が容易にできないといった問題点が顕著になってきている．

このような問題を解決する方法として，以下の3つの方式が考えられる．

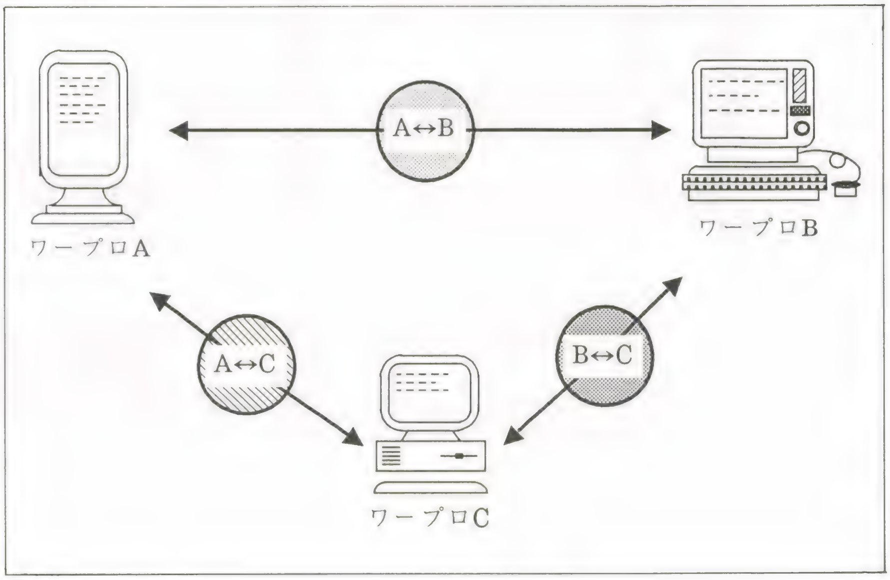
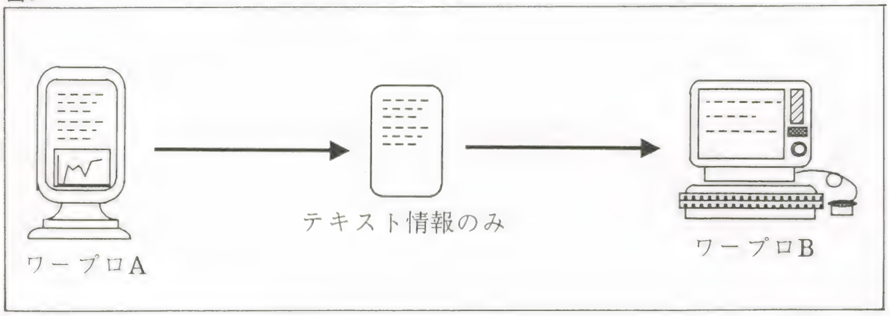


図1

（1）直接ワープロ間変換方式

異機種のワープロ間の変換ソフト（コンバータ）を開発し，たとえば，直接ワープロAからワープロBへの1対1変換を行なう方式である．この場合，ワープロの組み合わせの数だけコンバータが必要になる．**図1**は，3つの異なるワープロ間の変換を示す．

図2



（2）テキストのみの共有化

文書全体の共有化ではなく，文書の主体であるテキストのみを交換し，図や属性情報（文字修飾など）を無視する方法である．**図2**では，ワープロAで作成した図が文書交換後ワープロBで欠落している．

（3）共通文書フォーマット

文書の共有化や交換を行なうための中間的な共通文書フォーマットを規定し，各ワープロとその共通文書フォーマットとの変換を行なう方法である．この場合，変換をつかさどるコンバータの数とワープロの種類の数が同数となる．**図3**では，各ワープロを共通文書フォーマット“X”に変換して文書交換を行なう場合を示す．

図や属性情報を含むほぼ完全な形で文書交換を実現する方法は，上記(1)と(3)である．(1)を実現するためには，変換するほうのワープロと，変換されるほうのワープロの双方を熟知する必要があるが，これらワープロの多くは公開されていないのが実情であり，コンバータを実現することは容易ではない．また，新しいワープロが登場したり，既存のワープロがバージョンアップした場合は，すでに存在するワープロの数だけコンバータが必要となり，現実的な究極解とはならない．

上記(2)は暫定的な解として受け入れられはするが，図や属性情報が欠落するため，最終的な解とすることは難しい．

そこで方法(3)に注目する．上述のとおり，各ワープロと共通文書フォーマットとの間の変換をつかさどるコンバータの数が各ワープロの数ですむと同時に，新しいワープロの登場やワープロのバージョンアップに対する作業を最小限に留めることができるといった特徴を有する．

近年，上記(3)を実現する共通文書フォーマットがいろいろと議論されている．国際標準化として研究されているODAやSGMLを筆頭に，具体的な製品を狙った以下のようなCDFF，MDI，CDAなどが見受けられる．

CDFF (Common Document File Format)

ソニー(株)の主導による10数社のベンダで結成される「CDFFコンソシアム」によって構築された共通文書フォーマットで，CDFF文書の編集・出力機としてNEWSを用いる（今春，三菱電機㈱のエンジニアリングワークステーションMEシリーズもCDFFのサポートを発表した)．以下にCDFFの特徴を示す（詳細はp.164を参照)．

－国内の各種ワープロ文書とCDFF文書とのファイル変換

－CDFF対応のアプリケーション(編集ソフトなど）の整備

－各種出力機器（NEWS用プリンタ，電算写植機など）への出力プログラムの整備

MDI (Multi-object multi-lingual Document Intermediary)

エンドユーザーが文書処理をより効率よく行ない，各システム間で文書情報を相互にそれも簡単な手続きで交換することを目的とした富士ゼロックス(株)開発の異機種間文書交換フォーマットである．MDIを採用することによりエンドユーザーは，異なるベンダの複数の機種間で文書情報を交互に交換・再利用できるようになる．MDIは以下の特徴を有する（詳細は次号を参照)．

－マルチオブジェクトな文書要素を表現可能

－文書表現の能力差に依存しない双方向交換

－多国語の文書交換にも対応可能

－マルチオブジェクト文書の機能拡張に柔軟な対応

CDA (Compound Document Architecture)

CDA体系は，マルチベンダ環境でネットワークを介してアプリケーション間で統合化文書を交換することを可能とするDEC開発のアーキテクチャで，その主要な利点は以下のとおりであり，DECはDEC利用者への積極的なサポートを狙っている（詳細は次号を参照)．

－テキスト，グラフィックス，スプレッドシート，チャート，表，イメージが混在するデータ（複合文書）の汎用的な交換，文書合成機能

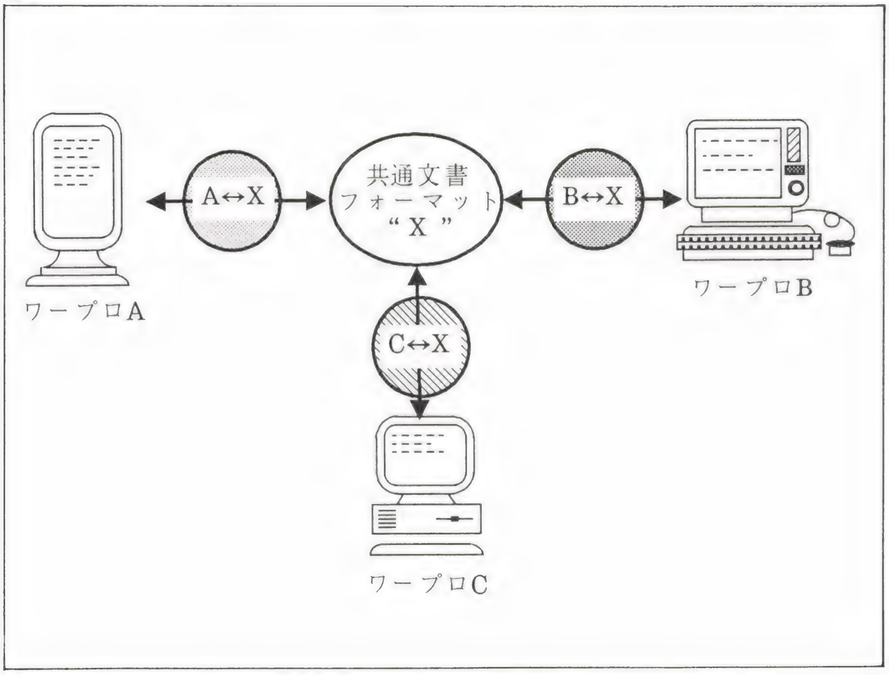
－複合文書の編集，転送，メール，表示，蓄積をネットワークを介して異なるOS間で実現

－CDAが提供する文書フォーマットと他の文書フォーマット（SGML，DCA，DIF，CGM，ODA／ODIF）とのフォーマット変換機能

－遠隔にあるリソースをLiveLinkを通して検索し，ローカルな文書に統合する遠隔アクセス機能（データの自動更新が可能）

本稿では，共通文書フォーマットの具体的な事例として上記3方式を詳細に順次紹介し，最後に筆者がこれら3方式と国際標準フォーマットであるODA（開放型文書体系：ISO8613）との関連についてまとめることとしたい．

図3



CDFF 共通文書ファイルフォーマット

ソニー株式会社

「CDFF文書処理環境」について

現在，日本語ワープロはあらゆるオフィスに普及し，文書作成に大きな威力を発揮している．しかし，異なる機種のワープロ間ではその文書ファイルの互換性はほとんどなく，「文書の共有化」や「効率的な文書処理」の観点からその対策が望まれていた．

ソニーでは，従来の文書処理がつねに抱えてきた問題点に着目し，「文書の共有化」と「効率的な文書処理」に主眼を置いて新たな「CDFF文書処理環境」を設定した．「CDFF文書処理環境」は，CDFFという統一した文書ファイル形式（共通文書フォーマット）を仲介として，**図4**のような文書処理環境を提供する．

この文書処理環境は，以下の3種類のソフトウェア群により構成される．

(1) 各種ワープロ文書とCDFF文書とのファイルコンバータ

CDFFを文書交換用の共通フォーマットとして位置付け，すべての変換をこのCDFFを介するようにする．また，CDFFは高度な記述能力を持っているため，文字修飾などのデータを捨てずに変換することができる．このように，多くのワープロ間の文書交換を容易にし，CDFF形式で保存することで，文書の一元管理も可能となるなど，効果的な文書の再利用が実現できるようになる．

(2) 編集ソフトなどのCDFF対応アプリケーション

CDFFを編集ソフトなどのアプリケーションの内部フォーマットとして直接使用することも可能である．この場合，同様なアプリケーション同士で（無変換で）文書データの交換が可能となる．

(3) 各種出力機器への出力プログラム

プリンタなどの出力機器に印刷する場合は，CDFFからその出力機器への出力プログラムを1つ作成するだけで，各種の文書の出力が可能となる．特に，電算写植機へのデータコンバータを作成することで，出版の領域までの文書処理環境が作成できる．

このような環境でNEWSをオフィスの核とし，これに既存のワープロやパソコンと接続することで，入力から編集，さらには写植機への出力までをトータルで扱うことのできる，従来では望めなかったような発展性をめざしたものがCDFF文書処理環境である．

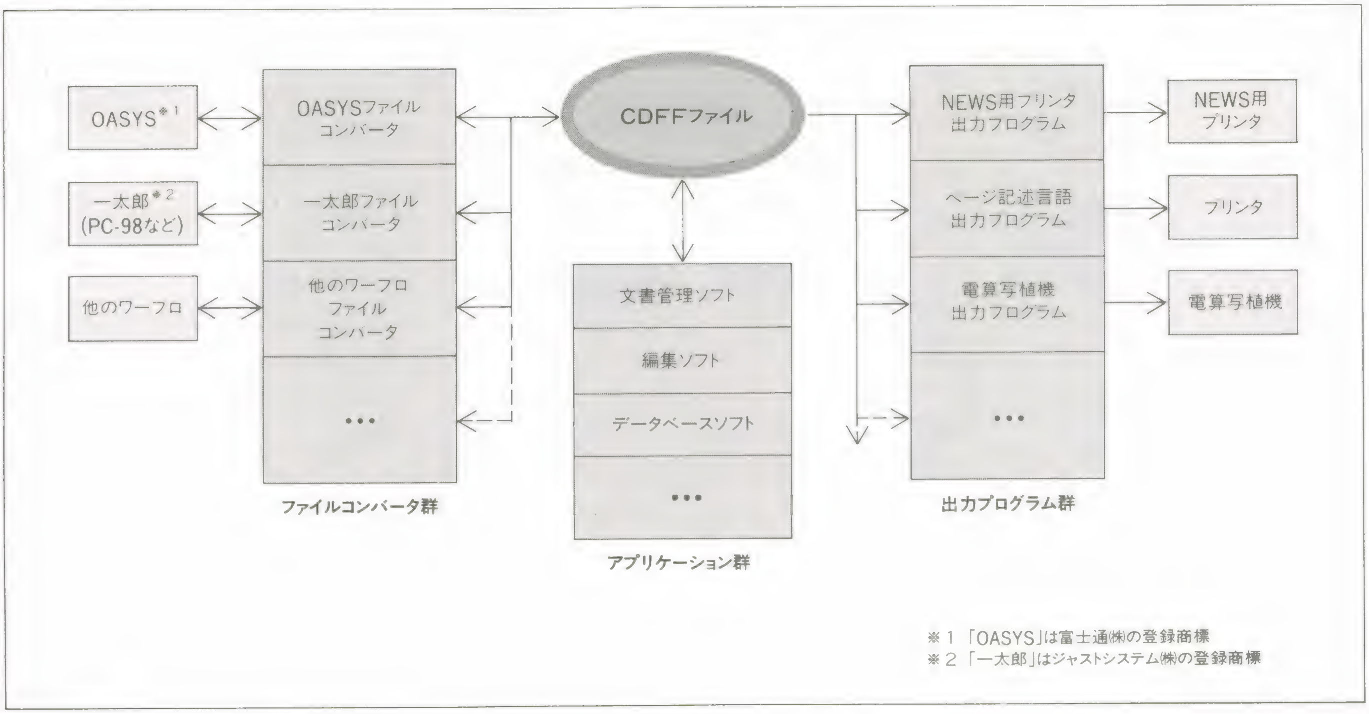
CDFFのフォーマット仕様

上記のコンセプトで開発したCDFFのフォーマット仕様を，文書構造，内容データ，割付構造，論理構造，ファイル構造に分けて以下に解説する．

文書構造

CDFF文書は，文書の物理的な位置付け・レイアウティングを表わす「割付構造」と，章，節といった文書の論理的構成の意味付けを表わすことのできる「論

図4　CDFFと周辺ソフトウェア群



理構造」の，意味の異なった２種類の階層を記述する構造を持っている．

さらに，文書の意味内容を記述する「内容データ」と呼ぶデータ群が先の２構造から指定され，これにより文書の割付，論理構成に対して内容が与えられ，１つの文書が構成される．

CDFFにおける文書構造を**図５**に示す．

内容データ

内容データは，上記の２構造から指し示される「意味」を持った独立したデータである．紙の上への現われ方以前の著者の意図が直接反映されたデータともいえる．

内容データの形式としては，現在以下の３種類が定義されている．

・テキストデータ：文字の並びとしての文書表現
・グラフィックデータ：ベクタ図形の表現
・イメージデータ：スキャナなどから読み込んだラスタ図形(画像)の表現

なお，このデータ形式は今後も追加・拡張される予定である．

各内容データは，さらにStream，Attributeと呼ばれる２種類の情報から構成される．

Streamは各内容データにおける主要な情報を，また，Attributeは対応するStreamへの修飾情報を保持するもので，AttributeはStreamに対する従属情報になっている．Textを例にすると，Streamは文字コード列であり，Attributeはそれに沿って書体・下線・網かけなどの属性を指定する一種のコマンド列となる．

テキストデータ

●Text Stream

TextDataのうち，文字そのものの情報を記述するデータ列であり，文字コードの並びとなる．CDFFで採用している文字コードは拡張UNIXコード（Extended UNIX Code-EUC）で，その中のコードセット３(外字用)には，CDFF独自に定義した外字セットが設定されている．

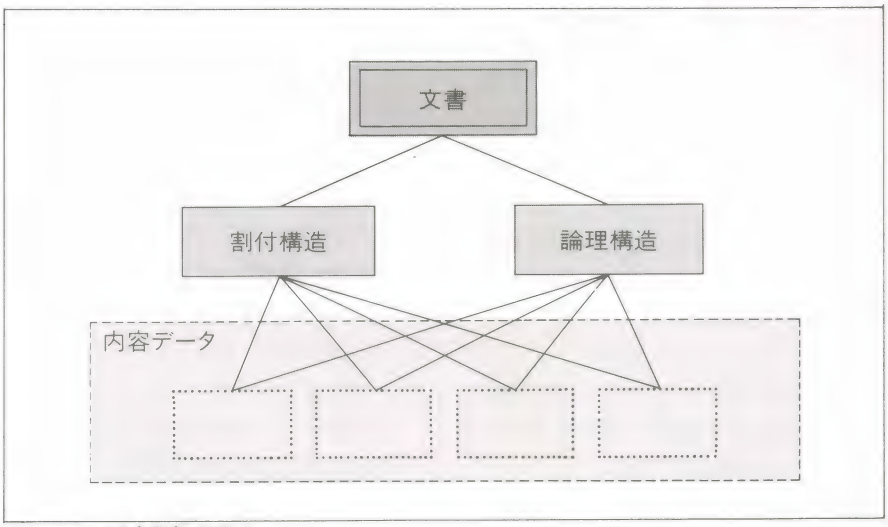


図５　CDFF文書構造の概要

●Text Attribute

Text Stream中の文字列に対する修飾属性を指定するもので，Text Attributeの基本構成単位は，12bytesからなるAttribute Entryと拡張Entryである．

各Entryと各フィールドの例を**図６**に示す．

なお，Text Attributeにより行なわれる文字列修飾の主なものは以下のとおりである．

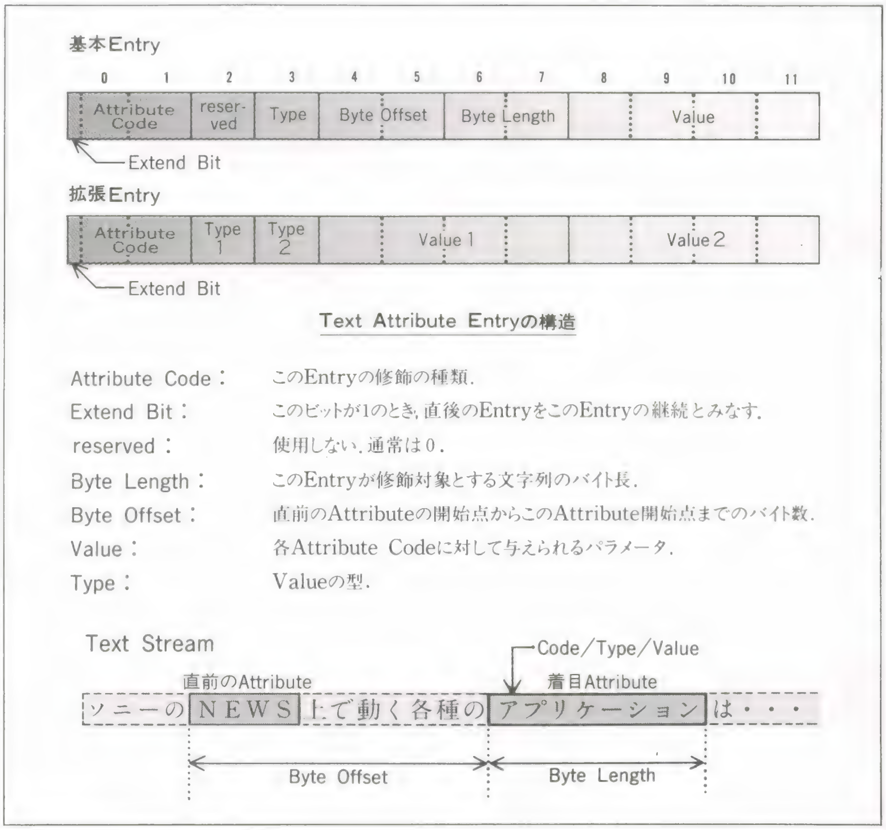
文字の形状／大きさ指定：Typeface，Point size，Scaling，Slantなど

文字の位置指定：Up/down，Character/Line，pich，Margine，Rotationなど

文字修飾の指定：Color，Under/Middle/Over line，Hatching，Outlineなど

流し込み指定：Line feed，Break page/frameなど

図６ 基本Entryと拡張Entry



グラフィックデータ

ベクタ図形を表現するデータで，線分／円などのベクタの形状を記述するGraphic Streamと，線幅／塗り潰しなどの情報を記述するGraphic Attributeから構成される．

●Graphic Stream

ベクタ図形の「形状（骨組み）」を記述し，基本構成単位は，オペレータの種類により可変長のGraphic Stream Entryである．その例を**図7**に示す．

CDFFのGraphic Streamのオペレータは，以下に示す種類がある．

ベクタ図形：Point，Line，Polyline，Polygon，Regular Polygon，Rectangle，Square，Lozenge，RoundedRectangle，Circle，Arc，Ellipse，EllipticArc，Fan，Elliptical Fan，Curve

その他：Group，String，Image，Define/Put Pict

●Graphic Attribute

Graphic Streamで指定された図形の“肉付け”を行ない，その構成単位はText Attributeと同じ12bytes長のEntryである．CDFFでは以下のAttributeを用意している．

Fill，Line width，Line Dash，
Logical operation，Color

イメージデータ

画像データ（ラスタ図形）を表現する部分である．Image Streamとしては，TIFF (Tag Image File Format)を採用している．なお，Image Dataに関しては，Attributeは設定されていない．

TIFFに関しては，“Tag Image File Format-Rev4.0”，An Aldus/Microsoft Technical memorandum，April 31，1987を参照．

割付構造

割付構造は，紙上に表現したときの文書の「レイアウト」を記述する構造であり，以下の記述をする．

・文書のページ構成
・段組構成
・各内容データのページ上の位置
・内容データ割付の諸ルール

割付構造は，以下の階層化された3つのオブジェクトで構成されている．

・Layout Root：1または複数の“Page”で構成される割付構造の根(root)
・Page：長方形の平面であり，最終的な展開出力先としての「紙」に対応．下位構造としての“Frame”の位置決めを行なう．
・Frame：1つの閉領域で，内容データを展開する「枠」．ここで，Frame内に展開すべき内容データか，さらに下位のFrameを指定する．

図8に，以上のオブジェクトからなる割付構造と実際の割付処理の一例を示す．

●Layout Entity

割付構造を記述する基本構成単位は，

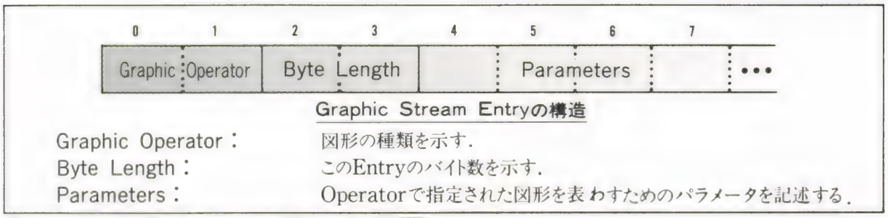


図7　Graphic Stream Entryと各フィールドの意味

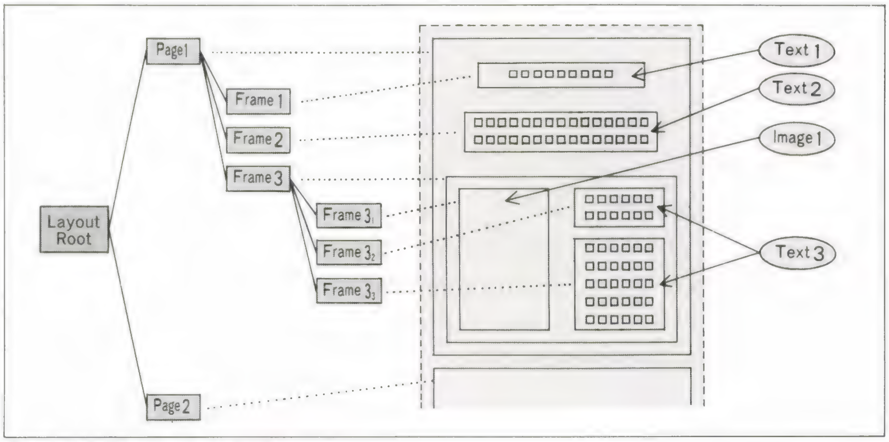
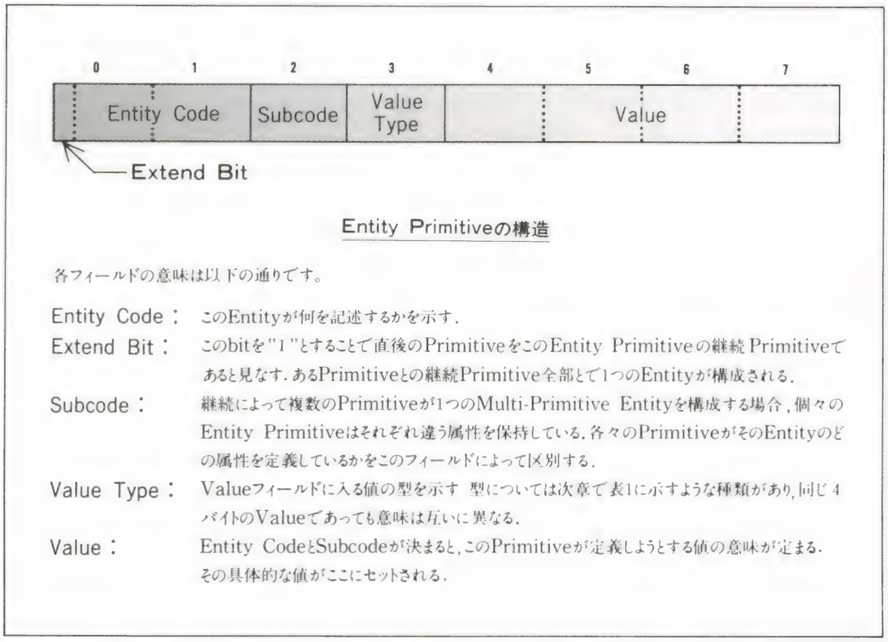


図8　割付構造とレイアウトの関係

図9　Entity Primitiveの構造



Entity名称	Place	意味	Entity名称	Place	意味
End	RPF	Layout root/Page/Frameの終了	Typeface	rpf	書体の指定
			Character Pitch	rpf	文字ピッチの設定
Page	R--	Page Dataへのpointer	Point Size	rpf	文字サイズの設定
Frame	-Pf	Frame Dataへのpointer	Character Scale	rpf	文字のスケーリング
			Character Rotation	rpf	文字の回転
Text Data	--f	Text Dataの指定	Slant	rpf	文字の斜体
Graphic Data	--f	Graphic Dataの指定	Color	rpf	色の指定
Image Data	--f	Image Dataの指定	Hatching	rpf	網かけ
Page Size	-P-	Page Sizeの設定	Underline/Midline/Overline	rpf	下/中/上線
Frame Shape	--F	Frameの形状の指定	Outline Character	rpf	枠のみ
Line Pitch	rpf	行ピッチの指定	Page No Format	rpf	ページ番号の形式
Line Direction	rpf	行の向きの指定	Line Width	rpf	線の幅
Writing Mode	rpf	縦書き，横書きの指定	Line Dash	rpf	ダッシュ線の指定
Top/Bottom/Left/Right Margin	rpf	各種マージンの設定	Logical Operation	rpf	オペレーション
Tab Position	rpf	タブ位置の設定	Define User Extchar	rpf	外字の定義
Right Justify	rpf	右揃えの有無	Define Tile Pattern	rpf	タイルパターンの定義
Newline Mode	rpf	自動改行条件の設定	Define Rule Line Pattern	rpf	ルールパターンの定義
			Define Dash Pattern	rpf	ダッシュパターンの定義

表 1　Layout Entity一覧表
"Place" の項はEntityの存在可能位置を示し，その意味は次のとおり．
第1桁：Layout Rootの記述に
R：必須，r：任意，-：禁止
第2桁：Pageの記述に
P：必須，p：任意，-：禁止
第3桁：Frameの記述に
F：必須，f：任意，-：禁止

図 9 の 8 bytesの固定長レコードである "Entity Primitive" である．

また，1つまたは複数のPrimitiveによって構成される，意味を持ったまとまりを"Entity"と呼び，割付構造は，Layout Entityのみで記述される．

なお，CDFF文書では，内容データを除くすべての記述はこのEntity Primitiveを用いて行なわれる．

また，**表 1** に実際のLayout Entityの一覧を示す．

論理構造

論理構造は，たとえば，目次，章，節などの内容データの意味上の構成・結び付きを記述することを1つの目的とした構造として考えられている．ただしCDFFにおいては，この論理構造に対して具体的（固定的）な意味付けや処理方法は実際には規定せず，割付構造以外の文書構造の記述手段を提供するに留めている．

これにより，事実上この論理構造は利用者に開放されており，各アプリケーションが自由にこの構造を利用できることになる．

論理構造は，以下の2つのオブジェクトから構成される．

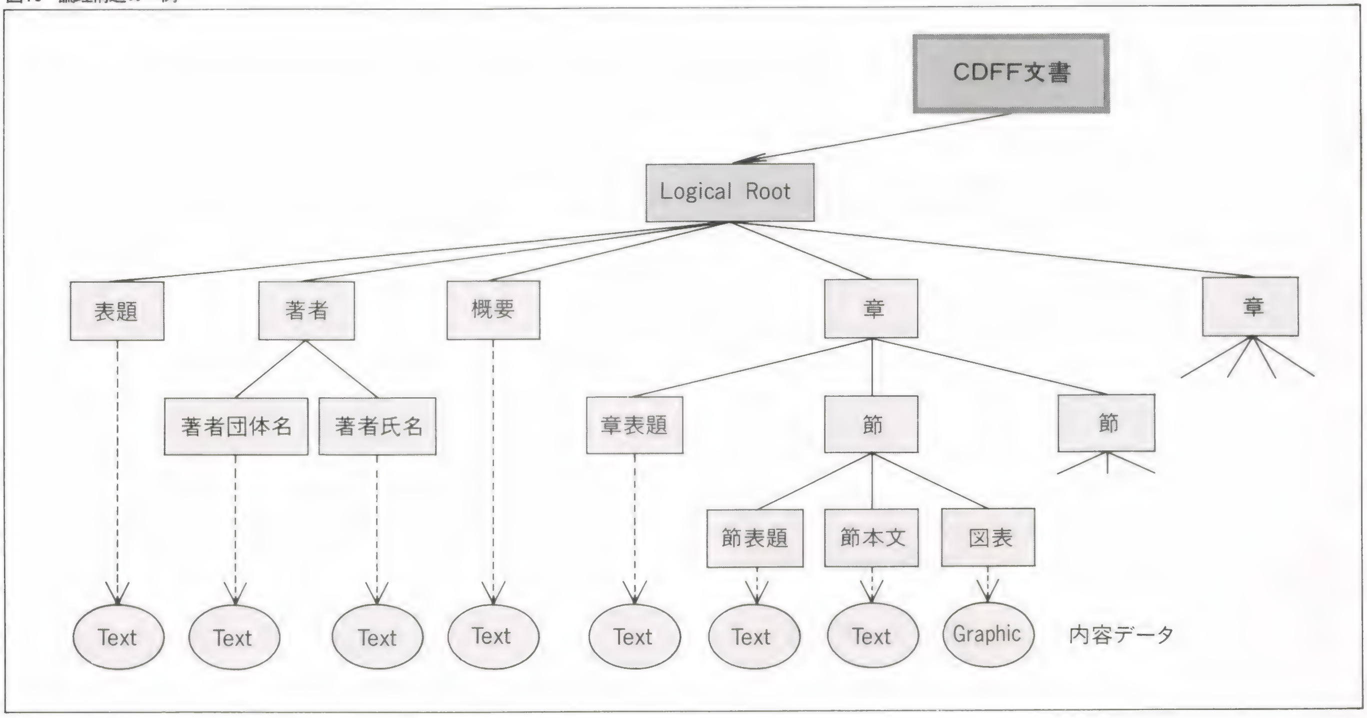
（1）Logical Root

1または複数の "Logical Node" から構成される，割付構造の根(root)である．

（2）Logical Node

論理構造を構成する実際のオブジェクトであり，この論理節に包含される内容データか，下位のLogical Nodeを指定する．

図10　論理構造の一例



以上のオブジェクトからなる論理構造の一例を**図10**に示す．ただし前述のとおり，CDFF自体では各論理節に“章”，“節”のような意味付けは与えていない．

●Logical Entity

Logical Entityも，その基本単位はEntity Primitiveによって構成される．**表2**にLogical Entityの一覧を示す．

Entity名称	Place	意味	Entity名称	Place	意味
End	RN	Logical Root/nodeの終了	Graphic Data	-n	Graphic Dataの指定
Logical Node	Rn	下位のLogical Nodeの指定	Image Data	-n	Image Dataの指定
Text Data	-n	Text Dataの指定	Free 1〜2047	rn	開放

“Place”の項はEntityの存在可能位置を示し，意味は次のとおり．
第1桁：Logical Rootの記述に　R：必須，r：任意，-：禁止
第2桁：Logical Nodeの記述に　N：必須，n：任意，-：禁止

表2　Logical Entity一覧表

ファイル構造

ここでは，CDFF文書がどのようなファイルによって構成されているかを説明する．

CDFF文書は，すでに述べたとおり，

・割付構造の記述
・論理構造の記述
・内容データ

の3種類の記述区分に大別できる．CDFFの文書は，1つまたは複数のファイルから構成されるが，CDFF文書の持つ記述区分のうち，割付構造の記述・論理構造の記述の2つは単一のファイル内にまとまって存在しなければならない（このファイルをCDFFのメインファイル，他のファイルをサブファイルと呼ぶ）．

また，上記の3種類の記述以外に，メイン・サブを問わずCDFF文書を構成する全ファイルの先頭にファイルヘッダと呼ばれる記述が必要である．ファイルヘッダはCDFFの処理系が文書を管理するために必要なもので，最低限，ファイル種別やそのファイルが従うCDFFのバージョン番号などが記述されていなければならない．メインファイルのファイルヘッダではこれに加えて文書名，著者名，作成日時など文書全体の管理用の属性値が記述される．

CDFF文書のファイル構成の一例を**図11**に示す．

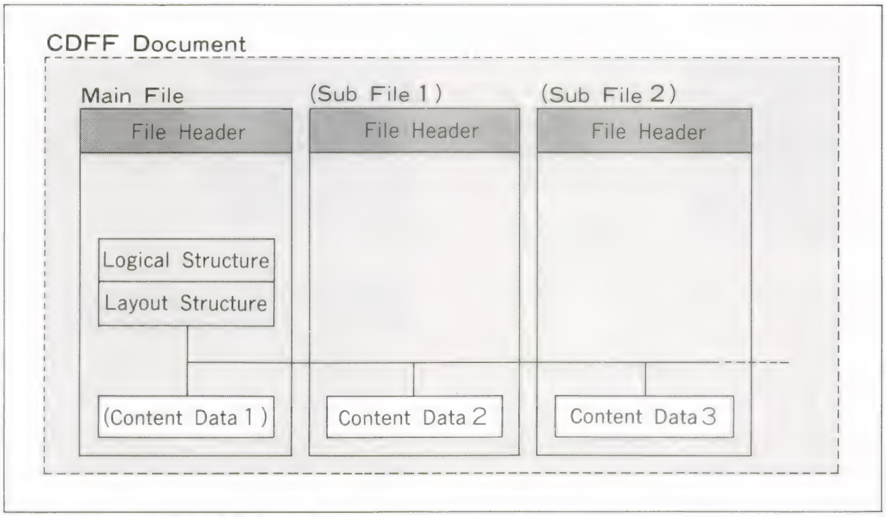


図11　CDFFの文書ファイル構造の一例

●ファイルヘッダ

ファイルヘッダの記述もEntityを用いて行なわれる．

ファイルヘッダ用のEntityを**表3**に示し，ここでの解説をしめくくることとする．

表3　ファイルヘッダ用Entity

Entity名称	Place	意味
CDFF version	MS	CDFFのバージョン番号
Measurement Unit	ms	CDFFにおける長さの基礎単位(解像度:BMU)の指定
Angle Unit	ms	CDFFにおける角度の基本単位(BAU)の指定
Document Name	M-	文書名を表わす文字列へのポインタ
Owner Name	m-	所有者名を表わす文字列へのポインタ
Creation Data	M-	文書の作成日時
W／P Name	m-	ワープロ機種名等の指定
Creator Information	m-	このCDFF文書を生成したソフトウェア識別
Revision	M-	CDFF文書の更新状況記録
Layout Root	M-	割付根の指定
Logical Root	m-	論理根の指定
Text Data	-s	Text Dataの指定
Graphic Data	-s	Graphic Dataの指定
Image Data	-s	Image Dataの指定

“Place”の項目はメイン／サブ両ファイルヘッダにおけるEntityのパーミッションを示し，意味は以下のとおり．
M：メインファイルヘッダに必須
m：メインファイルヘッダに任意
S：サブファイルヘッダに必須
s：サブファイルヘッダに任意
-：禁止

キーワード

EP環境への対応を迫られる文字,字体,字形,フォント

テレマティーク国際研究所
松下電送 小町祐史

“フォント”とは,統一的にデザインされた一連の文字およびパターン化記号のセットと定義されている.EP環境での情報伝達には,これらフォントを電子的に扱うための基本規格が必要となる.本稿では,前号での国際規格におけるフォント情報交換規格のあらまし,それに伴う用語の定義などを踏まえて,フォントデータ交換の国際規格を解説する.

フォントリソース規格

ISO/IEC JTC1/SC18 はフォントリソース(注1)の規格 ISO 9541 の開発を進めている.この国際規格については,

・“文字”に関して文化的,歴史的背景を異にする各国のあらゆるグリフ(注2)を扱える.

・世界中のグリフ集合のサブセットに適用されるあらゆるタイプフェイスに基づく多様なフォントを記述できる.

ことが求められている.

そこで規格9541はフォントリソースのデータ構造をそのアーキテクチャとして定め,数多くのフォント属性を定義している.さらに,それをSPDLのようなフォントユーザーに受け渡すための交換フォーマットと,形状表現データとを別パートに分けて規定している.現在作業中の規格案は次の3パートから成る.

9541-1, Architecture*[1)]

9541-2, Interchange Format*[2)]

9541-3, Glyph Shape Representation*[3),4)]

パート1,2の付属書(Annex)にそれぞれ記された,構造化名称,タイプフェイス・デザイングループも重要な規定である.

パート1,2については,今年の11月に第2次 DIS投票が行なわれることになっており,各国にレビューのためのドラフトがすでに配布されている.これらには,前回の投票における日本からのコメントはすべて反映されており,ルビ用フォントの属性,和欧混植のための属性なども定義されている.パート3のワーキングドラフトはSC18/WG 8 (SubCommitee18/Working Group8) 内に配布されて,内容の詳細審議を受けている.

アーキテクチャ

フォントプロパティ

フォントリソースは任意の数のフォントプロパティの集合として定義され,フォントリソース名で指定される.

ここでプロパティはフォント属性などのリソース情報を表わす要素データ構造であり,プロパティ名称,データタイプ,データ値を用いて記述される(注3).このプロパティはクラスに応じてまとめられ,プロパティリストとして関係付けられる.

構造化名称

ISO 9541規格では,命名手段として構造化名称 (Structured Name) を用いる.その理由は,さまざまな文書記述・処理言語で共有される,ほとんど無限に

注1) フォントの属性および形状表現データなどの集合を意味する.

注2) 意匠デザインを除いた文字形状の骨格表現であり,ほぼ字体に相当する.しかしたとえば1つのグリフに複数の字体が対応することもある.

文献* 1) J.D.Becker: “Unicode, A Proposed Standard Semantics for 16-bit Wide Character”, ANSI X3L2, 1989/09
2) ISO/IEC/DIS 9541-2, Font Information Interchange; Interchange Format, 1990/02
3) SC18/WG8 N1038, Proposed Working Draft for 9541-3, 1990/03
4) SC18/WG8 N1090, Editing Instruction for 9541-3, 1990/05

登録され続けるであろう公開オブジェクトに固有の識別子を与える必要性のためである．これは次のような特徴を持っている．

1）世界中の組織，オブジェクトをあいまいさなく指定する．
2）標準化名称およびプライベート名称の分散登録を可能にする．
3）登録済み名称を改訂せずにドメイン拡張を可能にする．
4）関連記述情報を付加できる．
5）直接的な名称など価性試験を可能にする．

したがってこれは，あらかじめ文字集合を規定してその要素に効率良く1つの識別子を割り当てる，JIS や ISO/IEC JTC1/SC2 の文字符号とはコンセプトを異にする．

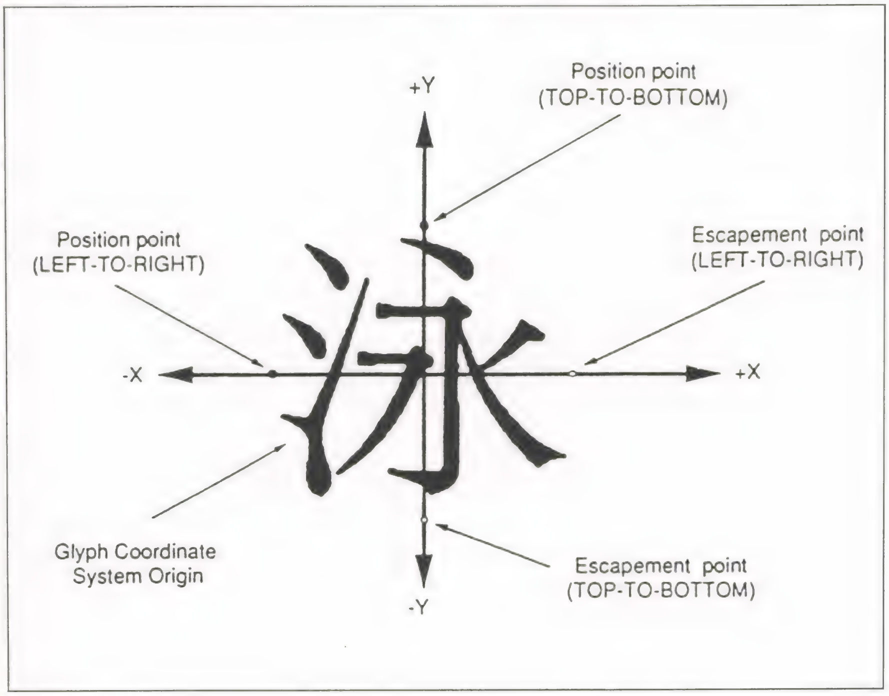
構造化名称は，オーナー識別子(owner-identifier) とオプションの関連記述メッセージとからなる．さらに，オーナー識別子は，ネーミングオーソリティ(**注4**)を階層構造で表わしたオーナー接頭語(owner-prefix)と，階層構造をとれる1つ以上のオーナー指定名称 (owner-assigned-name)とで構成され，オーナー接頭語だけが集中管理を必要とする．

オーナー指定名称はオーナー接頭語に導かれる名称記述文脈内であいまいさなく指定され，1文字以上の図形文字で表わされる．これを用いて，組織内階層ネーミングや多次元ネーミングを行なえる．

たとえば規格9541-1のオーナー接頭語は，「ISO(1) Standard(0) 9541(9541) 1(1)」のように表わされる．グリフもまた構造化名称で識別されるが，そのオーナー接頭語は，「ISO(1) Registration-Authority(1) 10036(10036) Glyphs (0)」であり，これに $1 \sim (2^{32}-1)$ の数値からなるオーナー指定名称が続く．

構造化名称では，1つのオブジェクトが同じまたは異なるネーミングオーソリティのもとに複数の名称を持つことができる．つまり同義語の存在が許される．しかしその生成をなるべく抑えるように，ネーミングオーソリティの管理が行なわれるべきであろう．

図1　漢字のライティングモード（LEFT-TO-RIGHTとTOP-TO-BOTTOM）



フォントリソース構造

フォントリソース・アーキテクチャは各種のライティングモード（**注5**）に対応可能である．多くのプロパティがライティングモードと関連があるため，リソース構造の中では，プロパティをライティングモード依存のModal Propertyと非依存の Description Property, Glyph Shape Representation Propertyとに分類し，Modal Propertiesをモードごとの値の集合にまとめている(**注6**)．インデントでネスティングを表現して，リソース構造を示すと次のようになる．

Font Resource Name Property
Description Property-list
Modal Property-list(s)
　Writing Mode Property-list
　　Glyph Metrics Property-list(s)
Glyph Shape Representation
　Property-list(s)

各プロパティの概要を次に示す．

Font Resource Name Property：ある1つのフォントリソースに含まれるすべての情報を参照するための識別子．

Description Property：1つのフォントリソースを，リソース全体として，ライティングモードとは無関係に特徴付ける．たとえば，Typeface Name, Font Family Name, Posture, Weight, Glyph Complement などである．

注3）たとえば，
weight-property : : =weight-name,
weight-value
weight-name : : =STRUCTUED-NAME
　-- ISO(1)Standard(0)954 1(9541)1
(1)/weight
weight-value : : CODE
のように用いる．

注4）ネーミングオーソリティにより，オーナー指定名称が定義される．ネーミングオーソリティとしては，ISO規格，ISO登録機関，ISOメンバボディなどがある．

注5）グリフ座標系におけるエスケープメント（送り）の方向（Position pointからEscapement Point への方向）．図1に漢字の場合の例を示す．

Modal Property：フォントリソースがサポートする各ライティングモードとそれに関するプロパティが，Writing Mode Property-listで指示される．

Writing Mode Property：グリフに共通する，ライティングモード依存のプロパティがリストとして指定される．たとえば，Escapement Class，Average Escapement，Maximum FontExtentsなどである．

Glyph Metrics Property：1つのライティングモードで定義される，各グリフの記述情報とメトリクス情報．たとえば，Glyph Name，Position and Escapement Points，Extents，Ligature Relationshipsなどである．

Glyph Shape Representation Property：後述「形状表現」で解説．

交換フォーマット

最小サブセット

国際規格に適合するフォントリソースとして不可欠なプロパティの集合を，最小サブセットとして，次の3グループに分けて規定している．

・最小フォント記述サブセット

Font Resource Name
Standard Version Number
Design Source Name
Font Family Name
Posture Code
Weight Code
Proportionate Width Code
Included Glyph Collection(s)
Included Glyph(s)
Design Size
Maximum Size
Design Group Codes
Structure Code

・最小モーダルメトリック・サブセット

Writing Mode Name
Nominal Escapement Direction
Escapement Class
Average x-Escapement
Average y-Escapement
Average Lower Case x-Escapement
Average Lower Case y-Escapement
Average Capitol x-Escapement
Average Capitol y-Escapement
Tabular x-Escapement
Tabular y-Escapement
Maximum Font Extent

・最小グリフメトリック・サブセット

Glyph Structure Name
X Escapement Point
Y Escapement Point
Extents

交換フォーマット

フォントリソースのデータは，さまざまな文書記述・処理系で利用される．そこでリソースデータをそれらに受け渡すための交換フォーマットは，ASN.1とSGMLの両方で定義されている．SGMLのDTD（Document Type Definition）による記述の一部を図2に示す．

図2 SGMLによって記述された交換フォーマット（部分）

```
<!-- Public document type definition. Typical invocation:

      <!DOCTYPE fontres PUBLIC "ISO 9541-2:1990//DTD Font Resource//EN">  -->

<!ENTITY % simval  "msg|str|int|card|code|propdata|relr|
                    ratl|bool|glbname|octstr|ang|time"                     >
<!ENTITY % compval "vlist|ovlist|plist|oplist"                             >

<!ELEMENT fontres    - -  (namencd? , namtabl? ,  -- 9541 font resource --
                      fontname , fontdes , wrmodes , gshapes? , niprop* )  >
<!ELEMENT namencd    - o (octstr)               -- char encoding for names -->
<!ELEMENT namtabl    - o (prefix , strucnm)+         -- name prefix table -->
                       -- see global name note at the end of this clause -->
<!ELEMENT prefix     - o (code)                           -- prefix index -->
<!ELEMENT fontname   - o (glbname)                            -- FONTNAME -->

<!-- Descriptive properties -->

<!ELEMENT fontdes    - o (dataver?  & stdver    & datasrce? & datacopy? &
                          dsnsrce   & dsncopy?  & relunits? & typeface? &
                          fontfam   & posture   & postang?  & weight    &
                          propwdth  & glycomp   & nomwrmde? & dsnsize   &
                          minsize   & maxsize   & caphght?  & lchght?   &
                          dsngroup  & structur  & mnfeatsz? & nomcsw?   &
                          nomlcsw?  & niprop*   )      -- FONTDESCRIPTION -->

<!ELEMENT dataver    - o (card? , card? , time?)            -- DATAVERSION -->
<!ELEMENT stdver     - o (card)                         -- STANDARDVERSION -->
<!ELEMENT datasrce   - o (glbname)                           -- DATASOURCE -->
<!ELEMENT datacopy   - o (msg)                            -- DATACOPYRIGHT -->
<!ELEMENT dsnsrce    - o (glbname)                            -- DSNSOURCE -->
<!ELEMENT dsncopy    - o (msg)                             -- DSNCOPYRIGHT -->
<!ELEMENT relunits   - o (card)                                -- RELUNITS -->
<!ELEMENT typeface   - o (msg)                                 -- TYPEFACE -->
<!ELEMENT fontfam    - o (str)                               -- FONTFAMILY -->
<!ELEMENT posture    - o (code)                                 -- POSTURE -->
<!ELEMENT postang    - o (ang)                             -- POSTUREANGLE -->
<!ELEMENT weight     - o (code)                                  -- WEIGHT -->
<!ELEMENT propwdth   - o (code)                               -- PROPWIDTH -->
<!ELEMENT glycomp    - o ( numglyph? , iglycols? , xglycols? ,
                           iglys?    , xglys?    , niprop* )
                                                              -- GLYPHCOMP -->
<!ELEMENT numglyph   - o (int)                                -- NUMGLYPHS -->
<!ELEMENT (iglycols | xglycols | iglys | xglys)
                     - o (glbname+)     -- INC/EXCGLYPHCOLS,INC/EXCGLYPHS -->
<!ELEMENT nomwrmde   - o (glbname)                            -- NOMWRMODE -->
<!ELEMENT (dsnsize | minsize | maxsize)
                     - o (ratl)               -- DSNSIZE, MINSIZE, MAXSIZE -->
```

注6）フォントリソース規格（9541）の中では多くのプロパティが規定されている．それらに対応する日本語表現に関して，まだJIS化委員会などでの検討は行なわれていない．したがって，混乱を避けるために，以降のプロパティについては，原文のままとする．

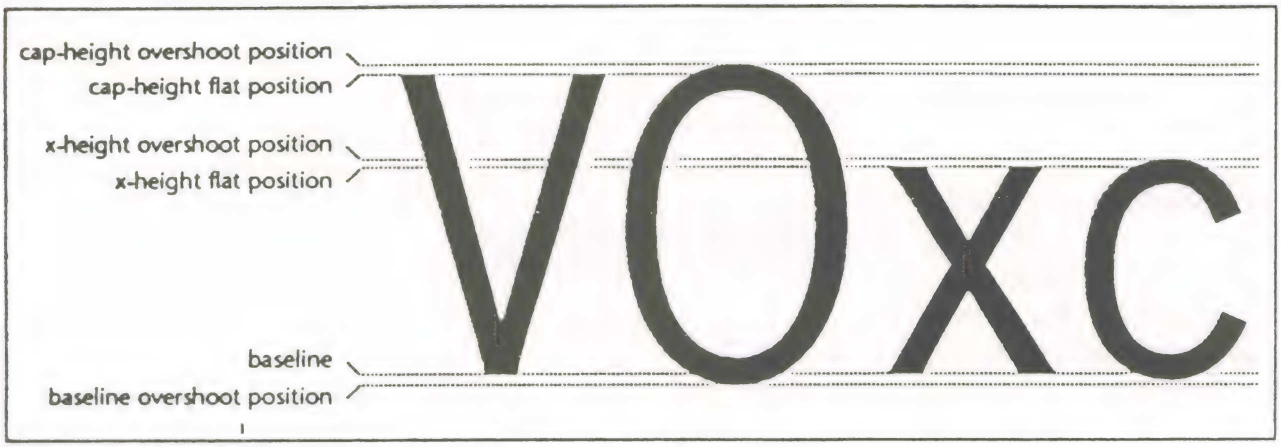


図3　3カ所のアラインメント

形状表現SPDLはType 1に決定か

経緯

今年の1月にSPDLの編集方針が変化し，“SPDLにグリフ形状表現の付属書を含め，Generic Fonts(すべての印刷出力に共通して使われる組み込み必須フォント）の規定を設ける”ことが要求された．これに急きょ対応するため，9541-3のエディタをDEC社のJim FlowersからAdobe社のT.O'Donnellに変更して，ワーキングドラフト(WG8 N1038)が作られた．その結果，この形状表現はAdobe社のType 1 Font Formatに近いものになっている．

日本と英国は，ラテン系グリフ中心のGeneric FontsをSPDLの規格に組み込むことに反対しており，特に日本は“Generic Fontsのようなコンセプトは SPDLの応用仕様(Functional Standard)の中で扱うべきである”ことを主張している．しかし米国は“SPDLによる文書交換の実用性を高めるために，Generic Fontsが不可欠である”ことを唱え，5月に配布されたSPDL改訂版と9541-3にAdobe色が強まった．

しかしこの議論はまだ終結したわけでなく，9月のSC18/WG8会議でさらに新提案が出されるはずである．形状表現フォーマットとして，日本は以前より，文字フォント開発・普及センターの高品質フォント標準化調査研究委員会の開発結果をSC18/WG8のフォント・プロジェクトに提案している．他国からの提案も含めて，9541-3の内容とSPDLの一部には，かなりの修正が加わることになろう．

ワーキングドラフトの概要

経緯に述べたように，ワーキングドラフトの内容がいまだかなり流動的であるため，ここではその概要構成を示すにとどめる．

思惑の焦点“ユニコード”見送り決定か

漢字符号に関連する最近のトピックとして，ユニコード(Unicode)*1)がある．これは米国の某有力ソフトウェアメーカーが中心となってANSIに提案し，さらにISO/IEC JTC1/SC2に承認を求めている文字符号体系である．ひと口でいうと，CJK(Chinese，Japanese，Korean)の主要な文字を2オクテットで表現しようとするものである．ユニコードの主な特徴は次のようなものである．

1)CJKの文字の中で頻度の高い文字を2オクテットで表現する．

2)C, J, Kの文字の中で“同じ”と判断されるものには，同一符号を割り当てる(いわゆるUnification)．

3)“同じ”と判断するための基準が，文字概念でなくて字体に置かれていることが多い．

図4にその例を示す．

特徴1)は多くの処理系を簡素化するため，ソフトウェア開発技術者の自然な要求に基づいている．この処理系の簡潔さこそが，ユニコードの基本思想であろう．特徴2)，3)は，特徴1)を実現するために2次的かつ必然的に生じた特性と言えよう．

しかし2)，3)は特に日本国内で多くの反論を巻き起こしている．その主旨は次のとおりである．

4)字体に基づく“同じ”文字であることの判断基準は，JISや従来のSC2とは異なる．

5)無限集合に近い文字群から使用頻度などに基づいて抽出されるべき符号化対象のサブセットの内容は，時間的変化をも含めて，C，J，Kの間で一致しうるものではない．単純なUnificationは，文字に伴う文化的，言語的な背景を無視することになる．

特に反論5)は，この符号系がユニコードという名称であることも手伝って，常に議論の中心になっている．

今年の4月にワシントンDCで開催されたSC2総会では，マルチオクテット符号化規格10646のDIS(Draft International Standard)化の問題とともに審議され，議

A．グリフ形状アーキテクチャ

・グリフ形状テクノロジー

本規格案ではアウトラインフォント・テクノロジーに基づくが，他のテクノロジーも許容する．たとえばType 1はGraphic drawing operatorとDeclarative hintsにより，形状を記述する．

・グリフ形状コンセプト

図3に示されるように，Flat PositionとOvershoot Positionが定義され，両者の間にAlignment Zoneが置かれる．Path, Paint Type, Sidebearing なども定義される．

・グリフレベル・ヒンツ

Declarative hints（ラスター画像に展開したときの品位を維持するための付加情報）の付加により，フォント展開処理系に水平ステムと垂直ステムの生成を指示する．

B．パーフォントリソース・データ

・一般プロパティ

Glyph Shape Technology, Glyph Procedure, PaintTypeなど．

・タイポグラフィックカラー・プロパティ

Apparent Weight（プリンタにおける視覚上の太さ）を制御する次のプロパティ：Top Alignment Zones（Blue Values），Bottom Alignment Zones（OtherBlues），Overshoot Suppression Size（BlueScale）など．

・プライベート・プロパティ

一般プロパティのGlyph Procedureに依存したプロパティ．

C．グリフ手続きオペレータ

D．サブルーチン

E．交換フォーマット

今後の展望―高品位文書の交換をめざして

フォントサービスへの対応

現状のフォントリソース規格9541にはかなり多くのプロパティが定義されているが，日本語の複雑な組版に必要なフォント属性が必ずしも十分に含まれているわけではない．たとえばルビ・アライメントライン上での各グリフの配置用属性，縦中横フォントの属性などは記述されていない．アラブ圏からは，Interglyph Connectionなどのアラビア語の記述に必要な属性の導入を要求されている．

このような言語依存の特殊表現，数学表現，およびマルチカラーへの拡張については，9541のパート4を設けてそこで規定することが，今年5月のISO/IEC JTC1/SC18総会で合意された．具体的な作業は9月からであるが，日本がかなりのイニシアチブをとることが期待されている．

各国語の特性への対応

SC18総会ではフォントサービスも審議され，これまでSC18/WG 4が開発を行なってきたDFR（Document Filing and Retrieval）をベースとして，DOA（Distributed Office Application）のクライアント・モデルに従った，新たなプロトコルと抽象サービスとを開発することが決まった．

送り手と受け手との間のフォントの一致までが要求されるような高品位文書の交換は，EP環境の拡大とともに増加するであろう．そこで求められるフォントリソース・データの転送には，このフォントサービス・プロトコルが不可欠であり，規格の早期完成が望まれる．（完）

事紛糾の末，投票に持ち込まれた．投票の結果，ユニコードの採用は見送られ，10646のDIS化を急ぐことになった．しかしUnificationのフィージビリティ・スタディがSC2総会以前に韓国から提案されており，SC2からJoint Research Groupに場を移して継続される可能性がある．

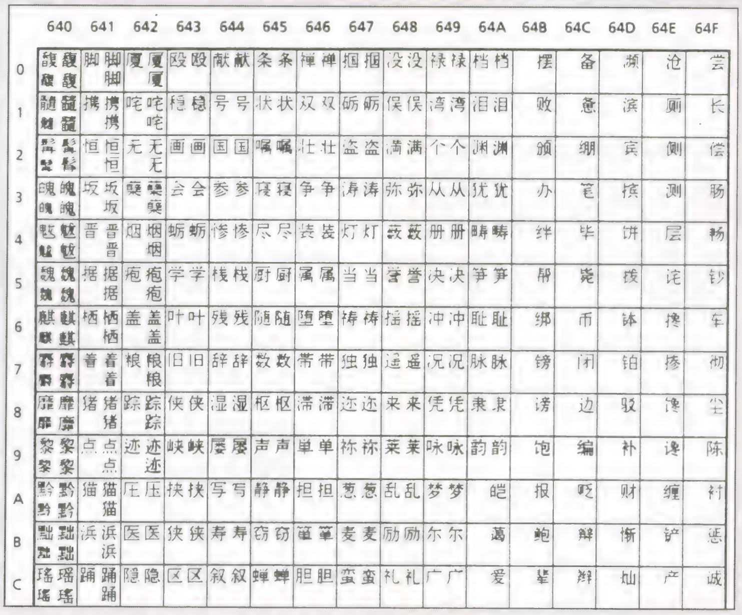


図4　ユニコードによるコード割り当て例

トピックス

世界初の"Digital World"レポート

マルチメディアの今後の展開への布石

ニュースページでDigital Worldの概要を紹介したので，ここではスピーチ内容について，昨年末日本で開催された初のマルチメディア国際会議'89との比較考察を交えて紹介したい．

"Digital World"とは

"Digital World"とは，Seybold氏によれば，コンピュータ業界，出版業界，エレクトロニクス業界はデジタルという領域で交わり，次第にその境界が消えていき，数年先にはデジタル領域で一体化する．そこで今これらの分野のテクノロジーが融合する世界としてDigital Worldについて話し合おうということである．氏はこの考え方をすでにSeybol dReportにも発表している．

では，そのテクノロジーの融合とは具体的には何かというと，現状では"マルチメディア"というところに落ち着く．

Digital Worldとマルチメディア

昨年12月半ばに，幕張メッセにおいて日本のハードウェアとソフトウェアのメーカー23社が参加して設立されたマルチメディアフォーラム主催による"マルチメディア国際会議'89"が開催された．

ここで，「マルチメディア」の定義が非公式にではあるが提案された．それによると，「情報を伝達する手段をメディアといい，コンピュータもメディアであると考える．そこで1つのコンピュータ上で，文字・画像・音声・動画などあらゆる技術を統合的に処理させるテクノロジーをマルチメディアテクノロジーと呼ぼう」ということである．この会議には，MITメディアラボ研究所長のNicholas. Negroponte博士，AppleフェローのAlan Kay博士，Apple社のJohn Sculley会長，ソニーの土井取締役，石田晴久東大教授といったこの世界のスターたち60人余りのスピーカーが参加した．

ここで，マルチメディアについて，その可能性，あるべき姿，何がプラットフォームとなるのか，などが語られたが，CD-ROMおよびCD-I, DVIがテーマとして大きく取り上げられた．しかし今回のDigital Worldでは，わずかにCD-ROMのビジネスについてOracle Data PublishingのDick Brass氏が熱弁をふるっただけで，CD-I, DVIあるいはCD-ROM XAといった特定の媒体に絞ったテーマを設けなかった．このことは，マルチメディアとして特定媒体に固執してしまう危険性を避ける意味でよいのではないかと感じた．

各セッションから

"Everything is Digital"は神話か

注目のS. Jobs氏は4人目のスピーカーとして登場した．彼は，Pixer社のCGを3本見せ，これがマルチメディアの集約された一つの姿だとしている．また，マルチメディアにかかわる"神話"として"Everything is Digital"があるが，これにこだわるべきではない，と述べたのが印象的であった．

T.NelsonはXANADU/Server発売を発表

この日のイブニングセッションでは，もう一人の花形T. Nelson氏が講演した．彼はコンピュータ分野での1960年代からのテクノロジーの推移について述べた．そして今マルチメディアあるいはヒューマンインターフェイスのあるべき姿としていわれているものはすべて彼が1960年にコンセプトとして提案していたこととオーバーラップしている，1990年の今まで30年間一貫してXanaduのコンセプトを追求している，と熱っぽく述べた．

ニュースページでも触れたが，そのXanadu/Serverを1990年末に発売する予定であることを発表した．

マルチメディアは教育には有効，しかし忘れてはならない愛情

教育にマルチメディアは取り入れられるか，で大学関係者（UCLAの教授など），メーカー，デベロッパー，そしてAND CommunicationsのR. Abelの6名のパネリストとSeyboldの司会で討議されたが，総じてマルチメディアは有効であるという事例が多く語られていた．具体的には，ロサンゼルスの4つの学校におけるコンピュータシステムの責任者というH. Kelly女史は，特に歴史教育で子供たちが興味を持つという効果が得られたと報告していた．

パネリストの1人IBMのLucy氏は，

IBMがAbel氏へ開発環境を提供したことを明らかにしたが，教育現場へのサポートから感じたこととして，特に小学生にコンピュータが非常に有効ではあるが，愛情を忘れてはならない，と指摘した．これは貴重な指摘として共感を呼んだ．

なお，R.Abelは新作“ユリシーズ”をプレゼンテーションした．前作“ゲルニカ”に劣らぬ迫力で感動を呼び起こし，プレゼンテーションが終わると同時に会場からは拍手がわき起こった．

ちなみに，この「ユリシーズ」は27日間で製作できたということである．

初日Evening Sessionで講演したTheodor Nelson（Autodesk社：右）とDick Brass（Oracle Data Publishing社：左）この2人一緒の質疑応答は2人の当意即妙なやりとりで会場をおおいに湧かせた．

ローエンドからハイエンドのDTCPを実現する「カラーポータビリティ」

イメージとカラーに関するEFI社のE.Arazi社長の講演は，会場を活気づけた．氏はカラー印刷業界の大手に成長したScitex社の創設者であるが，再度のチャレンジとしてEFI社（Electronics For Imaging）を設立した．目下，Adobe社のPostScriptレベル2におけるカラーのキーテクノロジーで，特許をめぐって係争中という渦中の人でもある．

人間の目がとらえるカラーの感覚とコンピュータとの融合など，カラーについてはいくつもの難題が未解決のままになっている．この「カラーポータビリティ」はそれらについてArazi社長がチャレンジした成果として開発された．これはデバイスインディペンデントなデスクトップ・カラー・パブリッシング（DTCP）を実現するためのソフトウェアである．

ニュースページで紹介したが，1990年12月にソースコードで出荷が開始されること，およびキヤノンのカラーコピアに搭載されることが発表された．

どうする？ データベースにおけるコピーライト

将来にそなえた現実的テーマとして著作権／知的所有権に関するパネルディスカッションは地味ながら重要であろう．ここでは，アプリケーション制作の際，所有権／著作権に関わるコストが非常に大きな壁となっている現状が述べられた．

Voygers社のR.Steinは一例として，あるフィルムの中の1カットだけしか使わない場合でも，全フィルムを借りる，という簡単な契約をするほうが，1カットだけの借りだしのために複雑な手続きをする法律家を依頼するよりも安くつく，と述べた．

また，ABC NewsのD.Bohmanは，マルチメディア・インタラクティブのアプリケーションである「AIDS（エイズ）」をプレゼンテーションしたが，この場合にも著作権／所有権で苦労したと報告．同社では一定期間のライセンス契約を結んで，オーディオCDの場合と同じ14％を支払っているという．

質疑応答で，会場からT. Nelson氏が発言した．その内容は，これからはデータベースでも著作権使用料など権利関係の手続きいっさいを含んだ情報が得られるようにすべきであろう，そのために彼のAutodesk社では特別な工夫を考案したが，その場合にもXanadu/Serverは非常に有効なツールとなっているという．このようなデータベースを1993年までに完成させるという．

“まとめ”はAldusのBrainerd社長

全セッションを通じてのまとめとして，Aldus社のP.Brainerd社長がビデオによるスピーチとプレゼンテーションをした．Brainerd社長は，情報は誰にどのように使われるのか，また情報が正確に使われているか，正確にアクセスできるのか，といった“コンテキスト”が大事であること，今後あらゆる人との間で情報交換が可能でなければならない，2020年には“effeciency（効果的）”“transpiarent（使いやすさ）”“personalize”（個人的）が統合されるべきである，と述べた．

Digital Worldの今後の展開

3日間を通じて，スピーカー全体が期待しているのは，マルチメディアのエンドユーザーへの普及である．そして，スピーカーが望ましいマルチメディア製品として共通して挙げているアイデアは，要するに，現状のテレビやビデオをインタラクティブに利用できるシステムである．マルチメディアを広めたいとしてここに集まったスピーカーがめざしているのは，そのようなシステムの実現と効果的なアプリケーションの制作である．

マルチメディアが今後，はたしてビジネスとなるかどうかは，プラットフォームのいかんにかかわらず，エンドユーザーをターゲットにしたアプリケーションの数が増えることであるが，そのためにはある程度のフォーマットの標準化が要求される．これに関して，7月半ばに米国で，MacroMind社を中心に関連9社が会議を開くという．マルチメディアの今後の発展はこのあたりの動向にかかっていると思われる．（沼尾禮子）

業界動向

コンピュータ化へ急シフトをする軽印刷

伸びゆく軽印刷展より

中小の印刷業者に向けた「第27回 伸びゆく軽印刷展」が7月6～8日の3日間，幕張メッセで開催され，10万6000人の来場者を集めた．主催は，東京都と東京軽印刷工業会，日本東京軽印刷資材連合会，(社)日本軽印刷工業会．

今回は，昨年までの晴海から幕張メッセに会場を移し，展示面積，出展社数ともに過去最大を記録している．内容も特にカラーPostScript関連製品や多数のコンピュータメーカーからの出展など，電子化のイメージを強調し，“DTPストリート”として設置した11社協力によるDTPシステムの展示・実演ブース入り口は，崩れた壁をデザインし，「ベルリンの壁は破れた!!」と，軽印刷業にも変革期が訪れている現状をアピールしている．この背景には，コンピュータ導入による印刷形態の急変，また人手不足といった印刷業全般が抱える問題が大きく影響している．

DTPへの根強い期待感と入力の多様化

DTPストリートに展示されていたのは，日本電気㈱「文豪8M/DTPモデル」，㈱モリサワ「Macintosh DTPシステム(ソフトはPageMaker)」，㈱クスダ「FX550SA」，ハイデルベルグPMT㈱「Compert」，㈱モトヤ「LASER7EX」，㈱キュワット「TA1」，リョービイマジクス㈱「SuperDTP」，住友金属工業㈱「SMI/EDIAN」，㈱ミック「iPRO-8000」，電字システム㈱/㈱東芝「DTP7000」，㈱写研「HITCAP-SC32」の11製品．

配布していたチラシでは，“いま，どうする？”と来場者に問いかけている．

このDTPストリート以外にも出展社のブースでいくつかのシステムが展示されていたが，松下電器産業の低価格DTPシステム「Z500」や電子組版システム「KX-Jシリーズ」が各所で見られた．

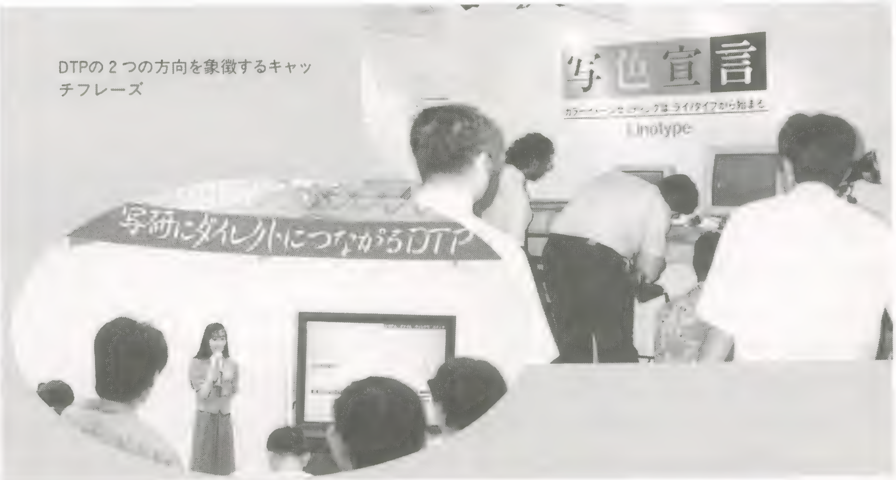
競合４社となった日本語PostScriptイメージセッタ

日本語PostScript(PS)対応のRIP(Raster Image Processer)は，すでにLinotype社と日本エーエム㈱(Varityper製)が投入しているが，今回日本アグファ・ゲバルト㈱が「AGFA9000PS MAX PLUS J」を初出展し，MONOTYPE社は，RIP内蔵型イメージセッタを日本語対応させた「PRISM PS PLUS KANJI」を出展した．

日本初公開のライノカラーシステム

日本語PSイメージセッタでも先鞭をつけたLinotype社が今回出展したのは，英語版PSによるカラー製版システム「ライノカラーシステム」．西独で開催されたDRUPA90で初出品したが，日本公開は今回が初めてとなる．

このカラーシステムは，Macintosh II fx，ライノカラースキャナRT600，ライノ20インチカラーモニタ，アップル13インチカラーモニタ，24ビットビデオカード，300Mbytes外部HDD，600MB光ディスクのハードウェア構成．これに出力装置として，同社のPostScriptRIP30，イメージセッタ330を組み合わせることで実現する．　(高木淳彦)



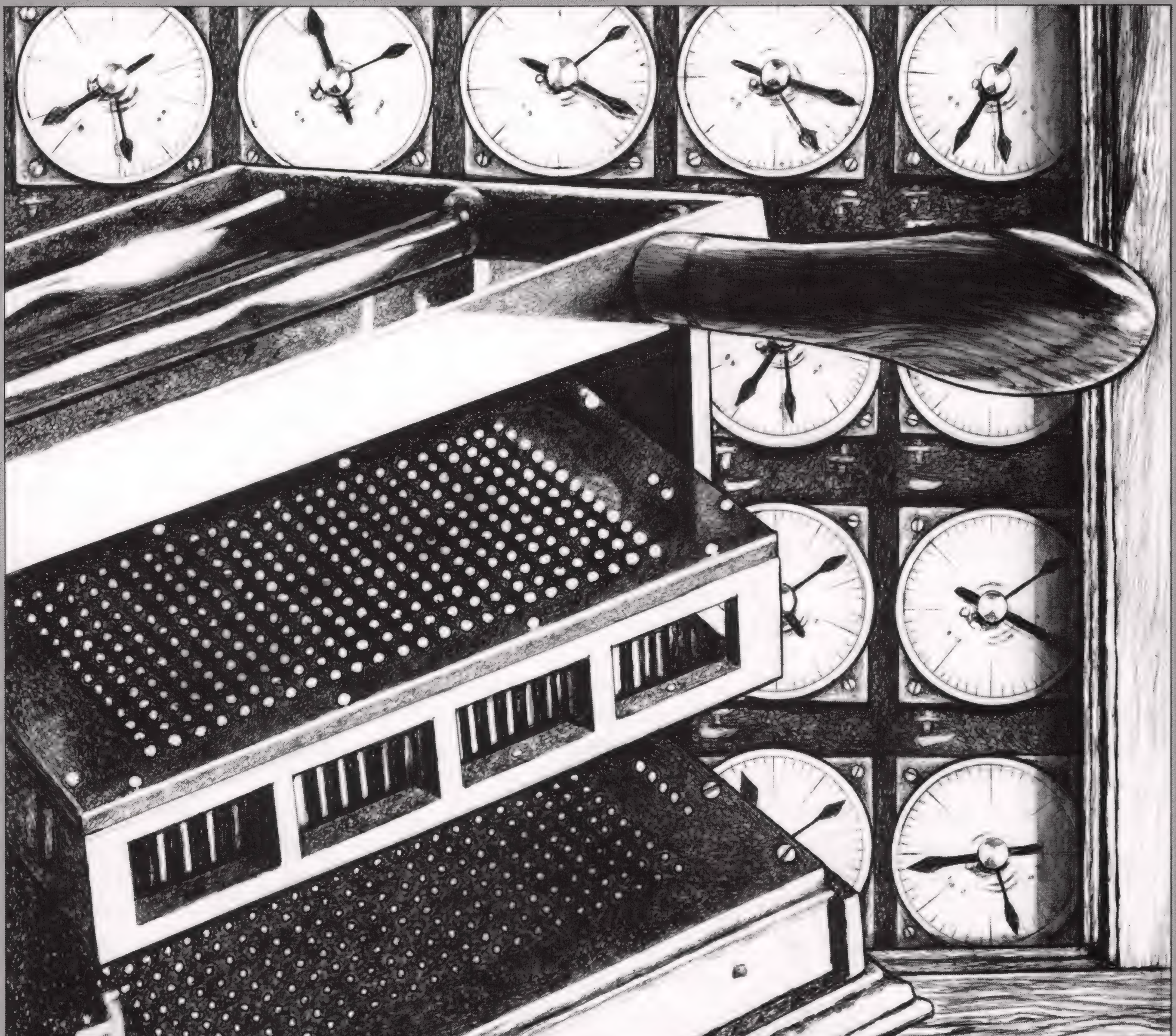
DTPの２つの方向を象徴するキャッチフレーズ

MIPS

MIscellaneous Packet Stream

パソコン1024夜
プログラマのための数字的世界観入門
ウサギと月と，そして太陽と
Fat Bits
BOOK SHELF

illustration/Fumio Kanno



1890年のアメリカの国政調査で使われたパンチカードによる集計機．H.ホレリスの手になるもの、かのIBMもこのタイプの集計機のメーカーとして生まれた．

第2回　日本の夏・メゾンの夏

福崎俊博

寅さんは実在した

相変わらず夏である．暑い．CRTに向かって原稿を書こうとしてもボ～ッとしたまま時間が過ぎ，ふと我に帰ると，仕事などそっちのけで夕食に何を食べるかを延々と考えている自分に気づく．クーラーのスイッチを入れれば暑さは解消されるのだが，電気代がもったいないという貧乏性と，冷房に頼りきりだと体温調節のきかない身体になってしまう気がするおかげで，クーラーをつけるのにもかなりの決心がいる．しかし今のようなマンション住まいでは，高温多湿の日本の夏をクーラーなしで過ごすのは単なる我慢大会と変わらなくなってしまうのがつらいところだ．

私がまだほんの子どもだった頃，家にはクーラーがなかった．それでも一応無事に夏を生き延びてきたのは，私が別に耐熱型の子どもだったからではなく，家が木造で風通しがよかったことと，扇風機，風鈴，ワタナベのジュースの素という「日本の夏の三種の神器」があったからだ．あいにくワタナベのジュースの素がないときには，別にスイカでもカルピスでもよかった．それに，家の前に出ている露店にいけば，かき氷がすぐに手に入ったのである．

葛飾区柴又にあった私の実家は，当時子ども相手の駄菓子屋を営んでいた．夏になるとどこからともなくテキ屋の人たちがやって来て，かき氷だのお好み焼きだのの露店をうちの店の向かい側に出す．そこで，シーズン前にはテキ屋の親分格の人が挨拶にきて，仁義を通していったものだ．映画の寅さんシリーズが始まる前の古い話だが，いくら柴又とはいえ，テキ屋の親分の格好がまるで寅さんそっくりだったのが今考えると面白い．そして私が10円玉を握りしめてかき氷を買いにいくと，親分は「前のお店のお坊っちゃまだ」と若い衆に声をかけて例外処理モードに移行し，皿の上にかき氷を普通の3倍くらい乗せてくれたのである．「お坊っちゃま」などといわれるのには子ども心にも閉口したが，親分が暇なときには一緒にセミを採って遊んでくれたりして，遠い夏の思い出として今でも鮮やかに心に残っている．

メゾンの夏

それから15年後，私はひと夏を丸々メゾンで過ごした．この業界で仕事をするようになって初めての夏である．「メゾン」というのは，執筆者やスタッフ，プログラマが快適に仕事ができるように，アスキーが借りているマンション（「メゾン○○○」）の一室のことだ．編集者が仕事の遅い執筆者に「ちょっと何日か集中して仕事しませんか」と半強制的に誘うケースが多いからといって，メゾンは断じて「タコ部屋」などではない．何しろ，メゾンに来れば，お金がなくてご飯を食べられないという不安から解放されるし，お風呂にだって入れるし，とにかく仕事はやり放題，マシンは使い放題，どうしても眠くて仕事ができなくなったら許されて寝袋か毛布にくるまって床の上で寝られるのだ．しかもメゾンは港区南青山にある．誰もが一度は住んでみたい南青山の高級マンションで生活できて，しかもお金がもらえるとなれば，若く優秀なプログラマが続々とメゾンに集まってきたのも別段不思議ではないだろう．が，みんな本当に喜んでメゾンにやって来たかどうかは，メモリフォールトが起こってよく思い出せない．

その夏，私は編集者として，文字どおり寝食をともにしてメゾンの人々と付き合っていた．編集者といっても，単に執筆者が書き上げた原稿を眺めていればすむわけではない．時には食事の手配をし，お酒を差し入れたりして，メゾンの生活を楽しくエンジョイしてもらう工夫をしなければならないのだ．食事の注文は，もちろんみんな「大盛り」である．もしも日本語のあまりできない外国人が注文の電話を脇で聞いていたら，日本人は「オオモリ」という食物を主食にしているものと思い込んだだろう．

当時のメゾンでは，夜中におなかが空いたら「牛丼弁当の大盛り生卵つき」を買ってきて食べるのがトレンディだった．一度食べてみれば分かると思うが，「生卵つき」というのが実にゴージャスな点なのだ．みんなの脳裏に牛丼が飛び交い始

める頃合を見計らって買いにいくのだが，夏の場合には重大な障害が存在していた．なんと夏の間は生卵をお持ち帰りできないのである(衛生上の問題らしい)．現在の牛丼屋はこのあたりがバージョンアップされているかどうか知らないが，ともかく当時はおなかの空いた勢いと，NMIをかけてやろうかという気迫で店員を圧倒し，心の中で強く念じて，運よく禁断の生卵を獲得できる可能性に賭けるほかなかった．生卵を手に入れなかったらメゾンで仕事をしながら楽しみに待っている人々に申し訳なくて，メゾンに戻りづらかったのである．

また，カップ麺の出来上がる時間をブザーで教えてくれるプログラムを即席で開発した人もいた．どの現場でも，カップ麺は欠かせない食糧である．したがって，この種のプログラムはなかなか世間の目にこそ触れないものの，メゾンに限らず，おそらくもっとも開発されることの多いプログラムではないかと思う．MS-DOSなんかで何も考えずにプログラミングすると，カップ麺ができあがるまでただ単に待ち遠しさが募るだけのプログラムになってしまうが，きちんとTSRにしてやれば，その間の時間もコンピュータで仕事ができる合理的なプログラムになるはずだ．どちらがいいかはそのときの食欲や気分にもよるだろうが，TSRのプログラミング入門的な題材としては実用的だし，もってこいだと思う．

メゾン日記

食べる話題ばかり書いてきたが，メゾンの生活の中心は何といってもプログラミングと執筆だ．『標準MS-DOSハンドブック』を始めとするいわゆる「黒本」のシリーズ，『Cプログラムブック』，『MS-DOS SOFTWARE TOOLS』，『年刊AhSKI！』の記事等々，あの当時メゾンの中から世に出ていった書籍やソフトウェアは数知れない．みんな優れた資質を持っているところにそうしたメゾンでの経験が加わって，一人前のプログラマや執筆者に育っていったのである．

そう，メゾンこそパソコン界の「虎の穴」なのだ．みんなが寝静まった昼でも，CPUの設計やコンパイラの最適化アルゴリズムについて延々と論争している住民がいる．メゾンの住民はお互い心中密かにライバル意識を燃やして技術力を争い，新参ハッカーは先輩住民からコーディングの甘さを容赦なく指摘される屈辱に耐えながら，胃袋とともに腕を鍛えていく．メゾンにきてからたくさん食べるようになったという人も少なくなかったが，メゾンで暮らし始めて技術力が大幅にアップした人もまた多かったに違いないのだ．事実，メゾンを「卒業」した人々は各方面でバリバリ活躍している．ただ，「虎の穴」の出身といっても，誰もみな心優しいナイーブな青年たちであった．

そんな当時の住人たちが書き綴った「メゾン日記」が，今もメゾンの書棚にひっそりと置かれている．もう書き継ぐ人のいなくなったメゾン日記を開いてみると，8個のキーを各ビットに対応させて一度に1バイトずつビットパターンで入力できるようにするハッカー御用達16進キーボードプログラムのアイデア，いにしえのSC/MPの話や牛丼の話題，編集者が書いた愚痴，8086やFortranに対する悪口雑言などにまじって，若い住人たちの心情が素直に綴られた文章を発見できる．それを今読み返してみると，ほんのわずかな気恥しさを感じるとともに，文章の背後にあるわけの分からない真剣さに打たれる．そして私はこの6年間の空隙を思い，ほろ苦い杯を重ねながら夏の夜を過ごすのである．………

その後のテキ屋の夏

さて，テキ屋の親分のその後である．一緒にセミを採って遊んだ数年後，私が中学生，高校生へと成長して親分の露店に遊びにいかなくなったのと歩調を合わせるように，親分は忽然と姿を消してしまった．噂によると，テキ屋の縄張り変更で，私の家のあたりからスワップアウトされてしまったせいだという．代わりに露店を開きにきた連中は，私の家に挨拶にも来ない．小さい頃から職人に囲まれて育った生粋の下町っ子である母にはこれが我慢ならなかったようで，「仁義を教えてやる」と文句をいいにいったところ，テキ屋の若い衆がえらく恐縮していたらしい．このようにして，下町でも次第に仁義がすたれていくのである．

セミの代わりにトンボを採るようになり，テキ屋の若い衆が暇そうな表情を見せ始めると，夏休みも終わりに近づいたことを知る．そして，あれからいたずらに年齢を重ねた私はあの頃の宿題よりも重い仕事をこなし，母のために何度目かのお盆を済ませると，長く暑い夏もやがて終わりを告げるのである．

数学的世界観入門

プログラマのための

コンピュータで乱数に見える数列を作るには、慎重に選んだ適切な規則性を与える必要がある。

斎藤 明

1 線形合同法

前回見たようにプログラムで真の乱数を作り出すことはできない．しかしながら一見乱数に見える数列を作ることはできる．このような乱数のように見える数を擬似乱数と呼ぶ．

さて，そこで前回予告したように

$$x_{n+1} = (ax_n + b) \bmod k \quad (n \geqq 0)$$

という数列を擬似乱数として用いることを考えてみよう．もしこの形で擬似乱数を生成できれば，プログラムはきわめて簡単になる．この方法を線形合同法という．プログラム1に線形合同法により擬似乱数を出力するCの関数の例を挙げた．プログラムそのものはきわめて簡単で，問題となるのは定数a,b,kおよび初期値x_0の選び方である．

2 定数と初期値の選択（kが2のべきの場合）

前回見たように各x_iは0と$k-1$の間にある整数なので，その周期はk以下である．したがってkは十分大きくとらなければならない．ところが，kが大きくともa,b,x_0を下手に選ぶと周期がkに比べてきわめて小さくなってしまう．たとえば$k=16$とすれば，可能な最大周期は16である．ところが，$a=7, b=0, x_0=1$とすると生成される数列は

$$1, 7, 1, 7, 1, 7, \ldots$$

となり周期はなんと2になってしまう．この例からa,b,k,x_0は上手に選ばなければならないことが分かる．

さて，まず$k=2^e$としよう．kを2のべきにとるとmodの計算をシフトで行なうことができ，演算が簡単になる．

最初に$b=0$の場合を考える．このとき数列は

$$x_{n+1} = ax_n \bmod 2^e \qquad (1)$$

となり形が少し簡単になる．この形の線形合同法を特に乗算型合同法と呼ぶこともあり，次の事実が知られている．

事実１ $e \geqq 3$のとき，(1)で生成される数列の周期は2^{e-2}以下である．一方

$$\begin{cases} a \equiv 3,5 \pmod{8} \\ x_0 \equiv 1 \pmod{2} \end{cases}$$

とすると，生成される数列の周期はちょうど2^{e-2}となる．

表1に$e=4, x_0=3$としaの値を変えた場合の数列およびその周期を挙げた．この表について事実1が確かめられよう．事実1より，$b=0$のときには2^eを法としても決して2^eの周期は得られないことに注意しよう．

次に$b \neq 0$の場合を考える．この場合の線形合同法を特に混合型合同法ということがあり，次の事実が知られている．

事実２
$$\begin{cases} a \equiv 1 \pmod{4} \\ b \equiv 1 \pmod{2} \end{cases}$$

のとき

$$x_{n+1} = (ax_n + b) \; mod \; 2^e$$

で生成される数列はx_0の値にかかわらず周期2^eを持つ．

例１　今$e=4, a=5, b=3$とおくと，a,bは事実2の仮定を満たす．そこで$x_0=1$として数列を生成すると

$$1, 8, 11, 10, 5, 12, 15, 14, 9, 0, 3, 2, 13, 4, 7, 6, 1, 8, \ldots$$

となり数列の周期は$2^4=16$となることが分かる．

3 定数，初期値の選択（kが一般の場合）

ここでは，kが必ずしも2のべきとは限らない場合について見てみよう．このときには話は少し複雑になる．

まず$b=0$のときを考えよう．この場合kが２のべきである場合と同様に，aをどのようにとっても周期がkに達することはない．では，最大周期はどの程度になるのであろうか．この話をするためには位数，原始元という言葉について説明しなければならない．今，互いに素な正整数a,kを考えよう．このa,kに対し$a \bmod k$, $a^2 \bmod k$, $a^3 \bmod k, \ldots$と$a^s \bmod k$を次々計算していくと，いずれ1が現われることが知られている．この初めて$a^s \bmod k = 1$となるsをkを法とするaの位数と呼ぶ．

例２　(1) $k=16$，$a=9$とすると，aとkは互いに素であり，

```
random(x0,a,b,k)   /* 初期値 x0 および定数 a,b,k より        */
int  x0,a,b,k;       /* 線形合同法にて擬似乱数を生成する。*/
{
        int  i,x;
        x  =  x0;
        for(i = 0; i <= N; i++){
                printf("%d\n",x);
                x = (a*x+b)%k;
        }
}
```

リスト1
N項目までの擬似乱数の列を出力するプログラム．実際に大きな値の乱数を取り出すために使うときには，(a＊x＋b)の値がintで取り扱える範囲を超えないように，引数を設定してやる必要がある．

表1　aの各値による数列とその周期

a	数列	周期
1	3, 3, 3, 3, ...	1
3	3, 9, 11, 1, 3, 9, ...	4
5	3, 15, 11, 7, 3, 15, ...	4
7	3, 5, 3, 5, 3, 5, ...	2
9	3, 11, 3, 11, 3, 11, ...	2
11	3, 1, 11, 9, 3, 1, ...	4
13	3, 7, 11, 15, 3, 7, ...	4
15	3, 13, 3, 13, 3, 13, ...	4

$$a \bmod k = 9$$
$$a^2 \bmod k = 81 \bmod 16 = 1$$

となる．よって16を法とする9の位数は2.

(2) $k=16$, $a=3$とすると，aとkは互いに素であり，

$$3 \bmod 16 = 3$$
$$3^2 \bmod 16 = 9$$
$$3^3 \bmod 16 = 27 \bmod 16 = 11$$
$$3^4 \bmod 16 = 81 \bmod 16 = 1$$

となる．よって16を法とする3の位数は4.

上の例のように，kを決めてもkと互いに素なaの取り方によりaのkを法とする位数は異なる．aを$1 \leqq a \leqq k-1$の範囲内で考えたときその位数が最大となるものをkを法とする原始元と呼ぶ．この原始元の位数を$\lambda(k)$で表わす．

例3　$k=16$とするとき，$1 \leqq a \leqq k-1$なるaでkと互いに素なものの位数は以下の表のようになる．

a	1	3	5	7	9	11	13	15
位数	1	4	4	2	2	4	4	2

よって3,5,11,13が16を法とする原始元であり，$\lambda(16)=4$.
$\lambda(k)$の計算については次の公式が知られている．

事実3　$\lambda(2)=1$, $\lambda(4)=2$,
$\lambda(2^e)=2^{e-2}$ ($p \geqq 3$のとき)
$\lambda(p^e)=p^{e-1}(p-1)$ (pが3以上の素数のとき)

$k=p_1^{e_1}p_2^{e_2}\cdots p_t^{e_t}$と素因数分解されるとき，

$$\lambda(k)=\mathrm{LCM}(\lambda(p_1^{e_1}), \lambda(p_2^{e_2}), \ldots, \lambda(p_t^{e_t}))$$

ただし$\mathrm{LCM}(n_1, \ldots, n_s)$は整数$n_1, \ldots, n_s$の最小公倍数
さて乗算型合同法における最大周期は次の事実で与えられる．

事実4　$x_{n+1} = ax_n \bmod k$

で与えられる擬似乱数列の最大周期は$\lambda(k)$であり，これは

(1) x_0とkが互いに素であり，
(2) aがkを法とする原始元である

ときに得られる．
$k=2^e (e \geqq 3)$の形のとき$a \equiv 3,5 \pmod 8$であればaはkの原始元であることが知られている．このことと$\lambda(2^e)=2^{e-2}$から事実2はすぐに導かれる．もしkが素数ならば$\lambda(k)=k-1$となる．したがってこのとき最大周期は$k-1$となり乗算型合同法を用いても周期は十分大きくとることができる．

次に$b \neq 0$としよう．このときにはkを最大周期にとることができ，しかも最大周期を与えるa,kの選び方も分かっている．

事実5　$x_{n+1} = (ax_n+b) \bmod k$

で得られる擬似乱数列が周期kを持つための必要十分条件は

(1) bとkが互いに素であり，
(2) $a-1$がkのすべての素因数の倍数であり，かつ
(3) kが4の倍数であれば$a-1$も4の倍数である

ことである．

この事実はx_0については何もいっていないことに注意しよう．(1)，(2)，(3)を満足していれば初期値にかかわらず最大周期[※1]が得られるのである．

例4　$k=24$, $b=5$, $a=13$とするとa,b,kは上の(1)，(2)，(3)を満足する．実際$x_0=3$として数列を生成すると

3,20,1,18,23,16,21,14,19,12,17,10,15,8,13,6,11,4,
9,2,7,0,5,22,3,...

となり周期は24である．
上の事実で$k=2^e$とすると事実2が得られることも分かるであろう．

※1 周期が大きければそれだけで満足がいく擬似乱数が得られたことになるかというと，そうはいかない．たとえば$a=1$, $b=1$, $x_0=1$とすると，生じる数列は0,1, 2, 3, ...となり周期はkとなるが擬似乱数であるとはとてもいえない．

参考図書

●脇本和昌 "乱数の知識" 森北出版
●Knuth "The Art of Computer Programming,Vol.2 -Seminumerical Algorithms" Addison-Wesley,
邦訳 渋谷政昭"準数値算法/乱数"サイエンス社

ウサギと月と，そして太陽と

第2回　ワークステーションなんて怖くない

砂原 秀樹

いやー．本場のディズニーランドはやっぱり楽しいですね．ともかく，客のほとんどが外国人なんですから!!

何を馬鹿なことを言っているんだって？　実は，ちょっと仕事でアメリカに行ってきまして，あちこちまわってきました．それで，特におもしろかったのが，国際会議の会場の近くにあったディズニーランドというわけでした．

まあ，この話は別の機会にするとして，今回は……

★

これだけ「ワークステーション」と呼ばれる計算機が氾濫してくると，あちこちでワークステーション導入というシーンが増えてくると思います．そこで，今回は「ワークステーションの買い方」という話をしてみようと思います．でもここでは，どこのメーカーのマシンが良いとか悪いとかという話をするわけではありませんよ．パソコンでも同じですが，単にワークステーションを買うといっても，マシンの構成とか，ネットワークはどうするのかなど，いろいろと検討をしなければならないことがあります．さらに，意外と忘れられやすいポイントとかもあったりして，マシンが到着したのに動かせないなんてことになることもあります．

つまり，今回はこういったお話をしようと思っています．

★

ワークステーションを導入するときに，まず最初に決めるべきことは，何にワークステーションを使うのかということです．これによってディスクの容量とかが大きく変わってきます．だから，何となく「ワークステーションと呼ばれる計算機がはやっているようだから１台導入してみよう」というだけで買ってしまうと，中途半端な構成になってしまいあとで後悔することになります．だから，文書処理に使うとか数値計算をするんだとか，しっかりとした目的を持って導入を決めるべきだと思います．

★

さて，目的が決まって予算が決まると(実は予算獲得が一番重要だったりしますが)，それに合わせてマシンの構成を決めることになります．

初心者（？）の人が陥りやすいミスは，ここですぐにMIPS値に目がいってしまうことです．特に，１台目を購入する場合一番重要なのはCPU自身のスピードではないのです．

では，何が重要かというとディスクの容量なのです．特に１台目のシステムの場合，ディスクが足りないことは致命的な問題となります．ですから，予算のうちできるだけ多くをディスクに割り当てるべきなのです．

★

たとえば，文書処理システムの１つであるTeXを利用しようと思うとフォントなどを置くために60Mbytes程度の場所が必要になります．また，X Window System (Release 4 の場合) のソースコードをすべて置こうと思うと何と100Mbytes以上の場所が必要になってしまいます(GnuEmacsは30Mbytes程度です)．

「ディスクはいくらあっても足りない」というのは，システム管理者の合言葉にもなっているくらいです．ですから，ディスクだけは十分な容量を用意しておいても損はありません．

それともう１つ，ディスクの容量を十分に用意しておくと，２台目以降のシステム導入時にディスクレスシステムが導入できるという利点があります．１台目のマシンにディスクが十分ないと，ディスクレスシステムを導入しようとしてもディスクが足りないということになってしまいます．

で，容量だけあればよいのかというと，そうでもないのです．Unixの場合，ディスクのアクセス速度は全体の性能に意外と大きな影響を与えます．したがって，アクセス速度の非常に遅い（正確には，書き込み時の速度が遅い）光磁気ディスク(いわゆるMOディスク)が安いからといって，ワークエリア用のディスクに使ってはならないのです．ただ，MOディスクは，読み出しのアクセス速度はそれほ

ど遅くないので，読み出し専用のファイル（たとえばフォントなど）を格納しておくために使うことができます．また，Removableであるという特性を利用してバックアップ用，特にソースコードの保存用に使うと便利です．というのは，磁気テープと違ってランダムアクセスが可能ですから，ソースコードをちょっと参照したいなどということが簡単にできます．

このように，ディスク装置を選ぶ場合でも，その特性を考えて用途に合わせて選ばなければなりません．僕の経験からいうと，お薦めするディスク構成は，

固定ディスク250Mbytes＋MOディスク

というものです．ただ，あとあとのことを考えると，固定ディスクは500Mbytes以上あるのが望ましいでしょう．僕が大学で使っているシステムでは，ファイルサーバに約2Gbytes＋MO，あとのマシンに100M～200Mbytesというディスク構成にしてあります．

ともかく，容量が大きくなればなるほどbit当たりの単価は安くなりますから，ディスク装置にかけるお金はケチらないで，大きなディスクを付けましょう．

★

次に重要なのがメモリです．どのワークステーションでも仮想記憶を使っていますので，メモリが不足するとどんなにCPUが速くても性能は格段に落ちてしまいます．ですから，メモリも十分に用意しておかなければなりません．最近では，4 Mbit DRAMが使われるようになってきていますので，最低でも4 Mbytesということになってきていますが，やはり8 Mbytesはほしいところです．本当は，16 Mから32Mbytesぐらいあると快適に動いてくれると思います．

それで，お金が余ればCPUを速くするということになるでしょう．

★

ところで，ワークステーションを導入するときに忘れがちなことが電源です．たいていのワークステーションは，アースピンの付いた3ピンの電源プラグを利用していますので，それを使えるようにしておかなければなりません．また，大きなファイルサーバを導入すると特殊なプラグを利用している場合がありますので，確認しておかないと，マシンは来たのに電源がなくて使えないという情けないことになります（実はこれ，僕が経験した情けない事件でした）．

さてまあプラグの話は，たいていの場合すぐに解決するので，たいしたトラブルじゃないのですが，電源で非常に重要になってくるのが電源容量の問題です．ワークステーションは，パソコンと違って比較的電力を消費しますので，よく調べておかないと，ワークステーションを使うとブレーカが落ちるなんてことになってしまいます．特に，多数のワークステーションを導入する場合は，この電源の問題を真剣に考えないと，大変なことになってしまいます．

特に注意が必要なのは，カラーのシステムとレーザープリンタです．これらのシステムは意外と電力を消費しますので注意してください（特に，レーザープリンタは注意が必要です．たとえば，LaserShotは最大9Aも消費します）．

★

もう1つ忘れやすいことが，ネットワークのことです．まあ，とりあえずワークステーションを1台導入する場合は，ネットワークは使わないことが多いようですが，1台のマシンを何人かで使いたいですよね．そこで活躍するのが，まわりにころがっているパソコンです．もちろん，RS-232Cで端末として使うこともできますが，たいていのワークステーションはそれほどたくさんのRS-232Cのポートは付いていません（Sunは2ポートです）．で，役に立つのがパソコン用のイーサネットボードです．ちょっと高いのですが，単なる端末以上の機能が得られるので損はないでしょう．

で，ネットワークの話ですが，ビル全体にネットワークをはったりするのは大変な作業です．が，とりあえず部屋の中で何台かを接続する場合，いちいち太いイーサネットケーブルをはったりせずに，Thinイーサと呼ばれる細いケーブルを使うと便利です．このときに，必要なケーブルとターミネータと呼ばれるケーブルの両端に付けるものを忘れずに用意しましょう．

それから，今後ビル全体にネットワークを拡張する計画がある場合は，マルチポートトランシーバを利用することをお薦めします．普通，イーサネットケーブルへワークステーションを接続する場合，トランシーバをイーサネットケーブルに取り付けて，トランシーバとワークステーション本体をトランシーバケーブルで接続します．普通のトランシーバには，1台の装置しか接続できませんが，マルチポートトランシーバの場合，4～16台の装置が接続できます．また，イーサネットケーブルを用意しなくても，マルチポートトランシーバだけでネットワークとして動作させることができますので，簡単にネットワークの設置ができるのです．ここで忘れやすいのが，トランシーバケーブルです．注文のときに，忘れないようにしましょう．

★

といったところで紙面が尽きてきました．ワークステーションの導入作業は，これだけではなくソフトウェアのことも考えなければなりません．次回は，そんな話をしてみましょう．

Fat Bits

第2回　もっともっと，Windows

山崎俊一

ニューヨーク午後×時

6月某日，ニューヨーク52nd streets. CBSやTime本社の少し先にEggHeadはある．EggHeadは，ソフト販売の全米チェーンだ．このNY 52nd店は，見かけは小さいが顧客に大企業が多く，全米でも売り上げのトップクラスだと聞く．

一歩足を踏み入れると，店内はWindows 3.0とそのアプリケーションソフトの洪水だった．ワープロではAmi Professional, Word for Windows, DTPではPage Maker，Ventura PublisherのWindows版も間もなく入荷予定だという．グラフィックスではMicrografx DesignerにCorel Draw，その他hDCのアクセサリ類やToolBookなど，入り口一帯がWindows関連ソフトで埋まっている．「えらい勢いじゃないか」と店員に尋ねると，It's taifuneだかhuricaneだとかいって，店員は指を3本立てた．これは，three digits，つまり週に3桁，百のオーダーで売れているという意味だ．

この程度の規模のソフトショップなら，ニューヨークに10軒くらいはある．全米なら数百軒だ．ということは，Windowsは1週間で数万本売れているという計算になる．Microsoftの公式発表では，発売後1週間で10万本を出荷したとか．話半分としても，そういうスケールで売れていることは間違いないようだ．

雑誌を見ても，Byte，PC-Magazine，PC-World，PC Computingなど，主要なコンピュータ雑誌では，どれもWindows 3.0が表紙を飾っている．特にPC Magazineには，数十ページに及ぶWindowsの広告特集がうたれている．

Microsoftは，今回のキャンペーンに150万ドルの巨費を投じたという．MicrosoftがWindowsにかける意気込み，そしてWindowsの明るい将来を納得させるには十分な効果があったと思われる．

そして，Windows 3.0を使ってみた

EggHeadで購入したWindows 3.0を，帰国するなり自宅のPCコンパチ機にインストールして使ってみました．確かに画面デザインはソフィスティケートされているし，付属のアクセサリも充実しています．でも，でもです．普段Macを使っている筆者にとっては，不満がたくさんあります．どちらを向いてもWindowsでお祭り騒ぎとあっては，批判的意見も並べておくのがマスコミの責任でしょう．ここで，それらを列挙してみることにします．

1.Integrity　つまりウィンドウデザインの統一性の問題．Windows 3.0は3次元ボタンやカラーパレットなどで見映えがよい．が，使い慣れるとかえって騒がしく，落ち着きに欠ける．筆者は，Macのほうが上品で完成度が高いと感じてしまいます（写真1）．

2.デスクトップ管理の限界　これもMacとの比較．MS-DOSのファイルにはsignature，creatorといった記述ができない．アイコンの役割もその範囲に限られ，Macの一貫したデスクトップ環境とはかなり距離があります．

3.286アーキテクチャの後遺症　当然ながらWindowsは，MS-DOS，つまり8086（あるいは286のリアルモード）の旧世界との連続性がその特色になっています．でもこのために，モードが3種類もあってデバイスドライバまで限定され，リアルモードとプロテクトモードを切り替えるには，CONFIG.SYSの書き換えを要することもあります．その点，最初からリニアアドレスの68000系のCPUはうらやましい．

4.画面仕様　Windows自体の問題ではないけれど，VGA仕様の640×480ドットの画面でマルチウィンドウは狭苦しい．DTPやグラフィックスを利用するには，最低でも1,024×768程度は欲しい．そういうモニタはたくさんあるけれど，まだ3.0対応のドライバは手に入りにくい．実用的には，カラーなぞは無用だと思います（もちろん，用途にもよりますが）．

MacとWindowsの根本的な違い

結局，Windows 3.0の評価はMacとの比較になります．Macに追い付いたなど

という人も多いようですが，基本的な違いが2つあると思います．

1番目．MacはXerox PARC直系のラブチャイルドとかいわれ，ToolBoxというGUIモジュールやHuman Interface Guidelineという哲学を備えています．一方のWindowsはDOSの延長です．Integrityなんて問うほうが無理なのかもしれません．

2番目．DOSにはすでに数千万人のユーザーがいて，それなりの常識，あるいは操作モデルがあります．問題は，“C>”プロンプトとGUIによるウィンドウ環境では，差が大きすぎるということです．たとえば，WindowsのメニューはMacよりも項目数が多く，機能も豊富なように見えます．が，これをMacユーザーが見ると，“小さな親切，よけいなお世話”で，重複があったり手順が違っていたりと，煩雑かつ非論理的に見えることもあります．その大半は，Windowsの設計がDOSプロンプトとキーボードコマンドのMS-DOS体系をなぞっているための，無駄や無理だと思えます．

これは，よい悪いの問題ではありません．Mac世界の価値基準は，おそらく知的ソフィスティケーションにあり，それがMacの品の良さ，美しさ，Aesthethyを支えています．が，それだけにユーザーに対しても一定以上の知性が期待されます．“Macは子供でも使える”といわれていますが，子供は決っして馬鹿ではありません．その直感力では，大人をはるかにしのぐ場合も多いのです．Windowsはその逆で，あちこちつぎはぎだらけ，泥臭いけど86系パソコンで使える，という位置付けでしょうか．

現在，未来

MS-DOSとIBM-PC/AT（およびその互換機）は時代遅れの低機能とかいわれながらも，いまだに主役の座を譲りません．それが逆目に出てPS/2が伸び悩み，OS/2の普及の兆しも見えないまま今日に至っています．一方では，技術でも哲学でも優れているはずのMacが意外にも伸び悩む．そう思ってみると，この5年間はメーカーの思惑が外れ続けた，といっても過言ではないでしょう．

さて，現在はともかく，未来はどうなるのでしょうか．Windowsは，90年代の主役として花開くのでしょうか．もしもそうなった場合，OS/2はどうなるのでしょうか．ここでは，いくつかの判断のポイントを指摘できるでしょう．まず1番目はCPUの高速化傾向．2番目は他のOS動向．3番目はラップトップおよびノートブックPCの普及傾向といったところでしょうか．

CPU自体は，一時SPARCなどのRISCやVLIW（MC68050？）などの新アーキテクチャが話題でした．が，486以降はCISCの逆襲とかいってインテル系が優勢．SunのSPARCもMIPSのR-xxxxもむしろ防戦模様，モトローラ68系は後退気味．CPUの力関係だけ見れば，基本的にMS-DOS，Windowsに有利と考えてよいでしょう．

高速化した386や486でMS-DOSでもないだろう，というのはもはや常識です．そのとき，個人のMS-DOSユーザーが一番移行しやすいのは，やはりWindowsでしょう．しかし企業ユーザーなど，ネットワークのある環境ではUNIX，OS/2，さらにNovell OSなども有力です．386や486パソコンではOSの多様化が予測され，これまでのように，いつでも誰でもMS-DOSといった，単一OSの時代とは様子が異なってくると思えます．このように複数のOS環境が入り乱れる中で，Windowsがかなりのシェアを獲得する，と考えるのが妥当かもしれません．

そして最近もっとも注目されているのがラップトップ，ノートブックPCの成長です．昨年は40%近い伸び率だったといわれています．対するデスクトップタイプは，せいぜい20%程度の伸びでした．

ラップトップ，ノートPCの9割以上にはDOSがバンドルされています．このような背景をもとに，MS-DOSの復権を思わせるような現象が起こりつつあります．ウィンドウ環境下のアプリケーションがそうであるように，アプリケーションの高機能化，大容量化が進む一方で，Microsoft Worksなどといった軽量ソフトが脚光を浴びています．今後は，ラップトップではMS-DOSで軽量ソフト，デスクトップではWindowsで大型ソフトといった使い分けがより明確化されるでしょう．そういう機能格差がなければ，デスクトップ機の存在価値はありません．

それでも，期待はしています

ずいぶん勝手なことをいってきました．それでも，Windows 3.0は売れています．でもそれはまだ，“本質的な売り”ではないようです．今後，MicrosoftがどのようにWindows戦略（そしてOS/2戦略）を進めていくのか．ハードウェア，ソフトウェアの動向はどうなってゆくのか，などなど．watchする必要があるようです．わが家でも当分は，Windowsのカラフルな画面が，Macの画面を見つめる筆者の頬を照らすということになりそうです．

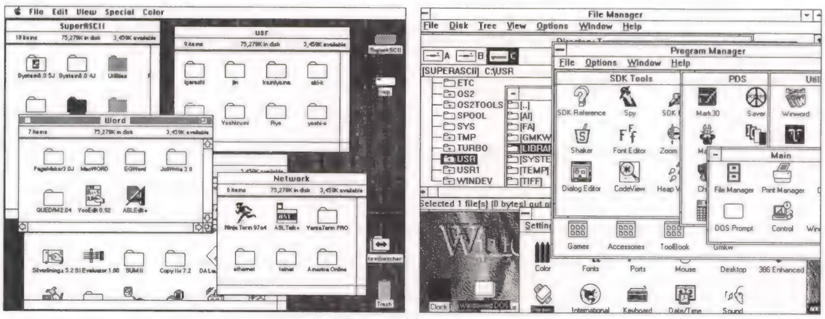
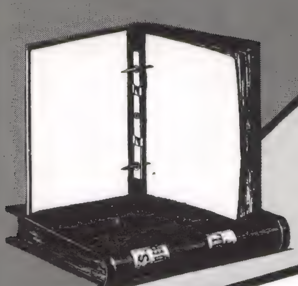


写真1　Mac II とWindows 3.0の画面



BOOK SHELF

KEEPING THE LINK
ETHERNET INSTALLATION & MANAGEMENT

Martin Nemzow

McGraw-Hill Book Company ,1988

出会いはぱっとしなかった

仕事柄，ネットワーク関連の書籍の購入には熱心なほうだと思っている筆者には珍しく，最初に本書を手にしたときの印象をほとんど覚えていない．

日頃出入りしている大学の研究室には，無造作に転がっていることが多かったが，この本を教材に輪講をやったという話は聞いたことがないし，学生との議論や雑談の中に登場したこともなかった．

筆者自身，黒い表紙のハードカバーの装丁の本書の存在は上記のような形で知っていたが，手にとって表紙を開くことはしなかった．

どうということのない本だと思ったのだろう，正直言って私の手元にある本書をいつどこで手に入れたのかすら覚えていないくらいなのだ．

だいぶひどいことを書いてしまったが，とにかく本書は，入手後しばらくの間は，机の上に積み上げられている未読の本のうちの1冊にすぎなかったのである．

しかし，数十台のバークレイ系のUNIXマシンを常日頃「さりげなく」管理する必要に迫られている筆者にとっては，『KEEPING THE LINK』というタイトルは，妙に気になるものがあった．考えてみると，このタイトルは多くのネットワーク管理者が日頃行なっていることそのものなのだ．というわけで，ようやく表紙をめくってみたところ，これが結構有用な本であることが分かった．

ひとことで言うとどんな本？

本書は，イーサネットを利用したUNIXネットワークの構築に必要なさまざまなノウハウを解説した本である．パソコンベースの世界だとネットワークの構築などと言うとまだ珍しいかもしれないが，バークレイ系UNIXの世界では，ネットワークは計算機環境の根幹をなす重要な機構である．ネットワークなしのUNIXマシンは，強いてたとえれば，ハードディスクなしのパソコンのようなもので，「まったく使えないわけではないが仕事にならない」のである．

話をもとへ戻そう．ひと口にUNIXネットワーク関連の書籍といっても，いろいろなバリエーションがある．その中でも，理論的な側面を紹介した本や，特定のプロトコルに関してプログラムを交えながら詳細に解説した本は数多く出版されている（少なくとも米国では）．

この系統の書籍はこれからも，この書評のコーナーで順次紹介されていくことになると思う．しかし，本書はこれらの書籍とは多少毛色が異なっている．

本書では，ネットワークを実際に設置するときに，現場ではどのような工具が必要で，どのような段取りで，どのような作業をすればよいのかを豊富な実例を交えて丁寧に解説してあるのである．

一例を挙げると，イーサネットケーブル（実体は同軸ケーブル）にコネクタを接続する際に必要となるケーブルの加工寸法とか，ケーブルのどこかで短絡事故が起こった場合（布団の中に針が紛れ込んでしまったようなもので，発見は難しいとされている）に，短絡箇所をテスタ1台で推定する方法といった，現場のエンジニアにはぜひとも知っていてほしい情報が，図解されている．「なんだ，じゃあメーカーのエンジニア向けの本じゃないか」と思うかもしれない．しかしこれは的を射ていない．ネットワークの設置

作業は，もちろんプロに依頼することもできるが，時間とお金を節約するため，しばしばユーザーが自ら作業する場合が多い．これは，お金に不自由していることが多い大学の研究室などでは当然のことと考えられている．しかし，できればこの手の作業はやりたくないというのもまた確かであり，結局このあたりのノウハウを持つ者に，常にお鉢が回ってくることになる．しかし，本書があれば新人に「教育的指導」と称して押し付けることさえ可能になる！ さらに付け加えれば，本書は図と写真が豊富なので，もし必要な情報だけ得られればよいというのであれば，「英語は苦手だからな」という尻込みも無用である．

構成を紹介しましょう

何やら，個人的な経験がちらりちらりと見えかくれする文章を書いてしまった．このままでは，なんのことやら分からないぞというお叱りを受けそうだ．これではいけない．ということで本書の構成についてもう少し具体的に紹介しよう．

本書は付録を含めて23の章から構成されており，全体が以下のような7つのPartに分けられている．

Part1 Overview
(1,2章)

Part2 Software and Hardware
(3章から8章)

Part3 Practical Aspects of Managing Ethernet
(9章)

Part4 Network Troubleshooting
(10章から14章)

Part5 Management Planning and Strategic Analysis
(15章から19章)

Part6 Troubleshooting Reference
(20章から22章)

Part7 Appendix

まずPart1「概論」では，ネットワークの有効性が論じられ，次にネットワークを理解するのに必要な一般的な用語や概念について解説が行なわれている．すでにネットワークの維持管理の経験のある読者には退屈かもしれないが，導入部としては適切な内容である．

Part2は，タイトルこそ「ハードウェアおよびソフトウェアについて」であるが，このタイトルから内容を予想すると良い意味で裏切られる．もちろん，OSIモデルやマンチェスタコーディングといった話題も盛り込まれてはいるが，イーサネットに使用するケーブルは，どのような材料を用いてどのように構成されているかとか，イーサネットケーブルを屋内に配線する際に必要となる用具（item）について，図を豊富に用いて説明している．たとえばケーブルを通すための樋（というか梯子）や，J型フック，BNCおよびN型の同軸コネクタの構造などが図解されている．さらに，同軸ケーブルを加工する際に，皮膜を何インチ剝き，どのように接続するのかといった，誰かに伝授してもらわない限り，自分ひとりでは決して完成できない面倒な加工方法も図解されている．トランシーバのタップの加工に関する解説にもページが割かれている．

Part3は一変してネットワーク管理にあたっての，管理者の心構えといった内容になっている．たとえば，マシンが落ちた際にユーザーからくる典型的な質問一覧などといったものがあり，管理者は常に答えられるようにしておくべきだと書かれている．どうやら洋の東西を問わず，システム管理者にくるクレームは同じ内容のようだ．

Part4では，ネットワークトラブルの修理方法について書かれている．ここではテスタ1台でどこまでネットワークトラブルに対処できるかが解説されていて，興味深い．筆者の経験からしても，十数台規模の小さなネットワークであっても，2,3年も運用していれば1度くらいはイーサネットケーブルの短絡によるトラブルを経験するもののようである．テスタの利用についてはは10ページほどしか記述がないが，要チェックである．

Part5では，ネットワークのパフォーマンスを向上させるための方法が論ぜられ，セキュリティ問題についても，その概要が論じられている．

Part6にはトラブルの際の対応がチェックリストの形で書かれている．どのような症状の場合には，どのような機材でどのような点をチェックしたらよいかが書かれている．もっとも，テスタしかないユーザーにはあまり役に立たないかもしれないが，参考にはなる．これとは逆に，ある障害が起きるとどのような症状が生じるかというチャートもあって，見ているだけでもおもしろい．

そして最後のpartは付録で，用語集と参考文献一覧が掲載されている．

なお，上記の各partの相互関係は，本書の前書きに図示されているので，本書をパラパラ流し読みするだけであっても，先に目を通しておくほうがよいだろう．

こんな人に推薦します

とりとめもなく書いてしまったが，とにかく本書は，これからイーサネットベースのネットワークを構築する必要のある人，あるいは，すでにネットワークを構築してきたが，職人的ノウハウを後進にどのように伝授したらよいのか苦慮している人には最適だと思う．なお，本書の邦訳はまだないようであるが，『UNIX magazine』に連載されている“UNIX communication notes”に，本書の内容と同様のアプローチで，ネットワークの構築方法について子細に記述した記事があった．参考のために付記しておく．

大野浩之
東京工業大学理学部情報科学科
計算言語学講座助手．
コンピュータ関連のテクノロジーに広範な興味を持ち，現在はコンピュータネットワーク，オペレーティングシステム，ヒューマンコンピュータインタラクションなどを研究テーマとしている．

読者からのお便りコーナー

新たに読者からのお便りコーナーを設けました．今回はスーパーアスキー創刊予告記事が掲載された月刊アスキー7月号のアンケートハガキ，その他に寄せられたメッセージの中からご紹介します．

From Readers

■SuperASCIIがどんな雑誌なのか，期待と不安でいっぱいです．ASCIIと両方買わないといけないとすると，また財政危機が訪れそうです．発売日をできたら3日くらいにするとかして欲しいですね．また，同時発売ならばSuperASCIIを買うとASCIIがついてくるなんてことはないでしょうか？　あとできたら，C MAGAZINE程度の薄さにしてほしいですね．今のASCIIは厚くて重く，カバンに入れて持ち歩く気がしないのですが…….

愛知県　ペンネーム　Apple River

◈

■SuperASCII，期待してます．ASCIIは新製品情報を含め，初心者向けに適当な雑誌であった．記事中には，多少高度なものやマニアックなものもありましたが，あまり目につかなかった．目立たなかった．これから脱皮したSuperASCIIは，初心者卒業者からは，おもしろい存在となると思われます．

愛媛県　神尾泰昭

◈

■むずかしい技術のことはよくわかりませんが，とにかく，トンガッた雑誌がまた一つ出来るとのこと．そこでお願い．オートデスク社サイバースペースプロジェクト関連，アップル社プロジェクト2000関連，BMUGの製品群について，テッド・ネルソン氏について，もちろんアラン・ケイ氏についても，またNeXTの日本語化の内容等々，以上，徹底取材して下さい．まだ，これらについて，自分を充分（納得?）させてくれた雑誌（記事）にお目にかかったことがありませんので……よろしくお願いします．期待してます．

京都府　平井浩政

◈

■「スーパーアスキー」，私には難しすぎるかもしれないけど，何やら面白そうですね．活字中毒の私としては，そのときそのときで一番興味のあることに関する本や雑誌をやたら読み漁ってしまう習性があります．そのため，必要以上に書籍代にお金がかかってしまうので，ぐっと我慢するように心がけています．(中略)「スーパーアスキー」，これは買ってしまいそうです．でも面白くなかったら，第2号からは買うのをやめるからね／　頑張って，面白い誌面にして下さい．

東京都　植杉伸介

◈

■以前の，おもしろかったASCIIがどこかへ行ってしまって久しいが，SuperASCIIには大いに期待している．願わくば「Super98」にならないことを祈っている．初心者向の記事もいいが，技術者向けの「難解記事」も希望したい．

神奈川県　中村隆生

◈

■スーパーASCIIなる物が創刊されるそうで，おめでとうございます．ASCIIから分岐した雑誌で，上位にあたる雑誌は初めてですね．内容の面では，現在のASCIIが初心者のパソコン取り扱いマニュアル的傾向が強まっていて，少々物足りなさを感じていたので，新しいASCIIに期待しています．しかし，何よりも素晴らしいのはSuperASCIIというネーミングのうまさについてですね．(中略)中学生の時から読んできたASCIIが一つ上にステップアップすることに，深い感慨を覚えます．

神奈川県　竹居正浩

◈

■うーむ，SuperASCIIか……　もし「お楽しみディスク」が付くとしたら，今度はさすがに3.5インチディスクだろうな……　いや，まてよ，対象が対象だけに「お楽しみストリーマー」なんてこともありうるかも？

岡山県　小田幸弘

◈

Return Mail

「スーパーアスキーがでると，いまのアスキーはアスキーオリジナルになるんですか」とか「スーパーアスキーの次は，コンビニアスキーがでるんですか？」などという社内では“おバカな”質問が連発するというのに，アスキーの読者の方々は，なんて善良な方が多いのだろうと，思わず目頭を熱くしながらお葉書を読んでしまいました．ということで，さっそく皆さんのお葉書へのお返事を書かせていただきましょう．

さて，創刊号，第2号と読んでいただいた読者の方には，すでにだいたいの内容はつかんでいただいたと思いますが，多くの方から寄せられた「技術者も読める……」「マニアックな」「難解記事」などという点では，なんとかご期待に沿うことができたのではないかと思います．今後ますます記事内容の幅を広げ，誌面の充実を図ろうと計画しております．安くて実用になるネットワークのつなぎ方，C言語やUNIX環境のツール類の渋い使いこなしなどなど，パソコン，ワークステーションを問わず，面白そうなネタをたくさん用意しています．次に京都府の平井さんのご要望されているテーマですが，創刊前というのになかなかスルドイところを突かれています．今月号のVirtual Realityの記事はいかがでしたか？最後に岡山県の小田さんのお手紙ですが，“お楽しみテープストリーマ”なんてことはありません．MOか9トラックテープです（ウソ）．(土屋)

AFTER CARE

1990年8月号

Super EXPRESS
P.58, 59の写真が入れ替っておりました．お詫びとともに訂正させていただきます．

Super EXPRESS
P.45の横河・ヒューレット・パッカード㈱の問い合わせ先電話番号が間違っておりました．

誤）03-330-6111
正）03-331-6111

関係者のみなさまには大変ご迷惑をおかけいたしました．謹んでお詫び申し上げます．

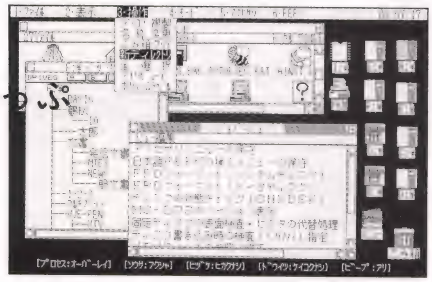
Super ASCII 読者プレゼント

Presents!

1

ですく・とっぷ

1名



(株)まつもとよりし，「でずく・とっぷ Ver.3.0」(PC-9800シリーズ対応).

2

オリジナルテレホンカード

30名

好評！　スーパーアスキー創刊記念オリジナルテレホンカード（50度数）.

3

MS-DOS SOFTWARE TOOLS 基本セット32

5名

ラクしてMS-DOSとつきあうための超便利コマンド集「MS-DOS SOFTWARE TOOLS　基本セット32」

4

Microsoft Worksの説明ビデオ

5名

Microsoft Worksのワープロ，データベース，通信，表計算の４つの機能を紹介する説明ビデオ.

読者プレゼント応募方法

読者プレゼントをご希望の方は，Super EXPRESS (P.33)の前にとじ込んであるアンケート・ハガキでお申し込みください．裏面に記入欄があります．希望番号と商品名をお書きください．応募者多数の場合は抽選になります．

なお，本誌記事に関するアンケートには必ずお答えください．プレゼント申し込みの締め切りは９月10日です．当選者の発表は，発送をもって代えさせていただきます．

スーパーアスキー編集部アルバイト募集のお知らせ

スーパーアスキー編集部では，アルバイト(編集サポートおよびプログラミングスタッフ)を募集しています．以下の要項を参照して，ふるってご応募ください．

■資格：18歳から25歳位までの学生の方で，週に２～３日以上当編集部で勤務できる方.

■待遇：時給700円以上，交通費支給.

■締切：1990年８月末日.
おって面接日などをお知らせ致します.

■応募方法：履歴書（面接日，連絡方法の希望があればそれも明記）および「私の欲しいコンピュータ」(600字程度．なるべくワープロ原稿で）を必ず封書にて，右の宛先までお送りください.

〒107-24　東京都港区南青山６-11-１
スリーエフ南青山ビル
(株)アスキー　人事部　「スーパーアスキー編集部アルバイト募集」係

最新ソフトウェアバージョンリスト

Software Version List

システムソフトウェア，ソフトウェアツールなどの最新動向が分かる，プログラマのためのバージョンリスト

	名称	バージョン	ベンダー	対応機種	リリース	備考
	SunOS	4.1	サン	SPARC Sun-3，Sun-4	1990年4月	SunOS 4.0とバイナリ互換 System V Release 4とソース互換
	SunView	1.8	サン	SPARC Sun-3，Sun-4	1990年4月	
	OPEN LOOK	1.0.1	サン	SPARC Sun-386i Sun-3，Sun-4	1990年4月	
○	HP-UX	7.3	YHP	HP Apollo 9000	1990年7月	HP 9000 S300，S800用はVer.7.0
○	DOMAIN-OS	SR10.3	YHP	Domain, HP Apollo 9000	1990年7月	モデル425，433，他のモデル用はSR10.2
	motif	1.0	YHP	HP9000	1989年8月	
				Domain	1990年1月	
	UniOS-U	1.3	オムロン	LUNA LUNA-GW LUNA-FS	1989年11月	統合化UNIX
	UniOS-B	1.5	オムロン	LUNA LUNA-GW デスクトップ	1990年2月	UNIX4.3BSD
	UniOS-Σ	1.3	オムロン	LUNA-Σ	1989年11月	ΣOS（V0R1）準拠
	Mach	2.5	オムロン	Luna-88K		1990年8月末に出荷予定
○	MS-DOS	3.3C	日本電気	PC-9800	1990年7月	メモリは384Kbytes必要
○		3.3C	日本電気	H98 model70	1990年7月	
		3.1	日本電気	98LT	1986年10月	
		4.0	日本IBM	IBM PS55	1989年5月	日本語版，英語版は1988年8月発売
		4.0	マイクロソフト	IBM PC，PS/2	1988年8月	
	MS-WINDOWS	2.11	日本電気	PC-9800	1989年10月	日本語MS-DOS Ver.3.1が必要
		2.1	日本IBM	IBM PS55	1990年5月	日本語版
		3.0	マイクロソフト	IBM PC，PS/2	1990年5月	
	MS-OS/2	1.1	日本電気	PC-9800	1989年10月	メモリは4.5Mbytes，要HDD（20Mbytes以上）
		J1.2	日本IBM	IBM PS55	1990年3月	日本語版，英語版は1990年5月発売
		1.1	マイクロソフト	IBM PC，PS/2	1988年8月	
	MS-C	5.1	マイクロソフト	PC-9800	1988年8月	Ver.6.0は秋に出荷の予定
		6.0	マイクロソフト	IBM PC，PS/2	1990年4月	
	MASM	5.1	マイクロソフト	PC-9800	1988年9月	
		5.1	マイクロソフト	IBM PC，PS/2	1988年7月	
	QuickC	2.0	マイクロソフト	PC-9800	1989年10月	
	QuickBasic	4.5	マイクロソフト	PC-9800	1989年11月	
	Turbo-C	2.0	ボーランド	PC-9800	1988年9月	MS-DOS 2.1以上，メモリ640Kbytes
		2.0	ボーランド	IBM PC，PS/2	1988年8月	PC-DOS 2.0以上，メモリ640Kbytes
		2.0	ボーランド	各種AXマシン	1989年8月	MS-DOS 2.1以上，メモリ640Kbytes
		2.0	ボーランド	FM-R PanacomM	1989年12月	MS-DOS 2.1以上，メモリ640Kbytes
	Turbo-C++	1.0	ボーランド	IBM PC，PS/2	1990年5月	PC-DOS 2.0以上，メモリ640Kbytes，要HDD
	Turbo-Pascal	5.5	ボーランド	PC-9800	1990年1月	MS-DOS 2.1以上，メモリ640Kbytes
		5.5	ボーランド	IBM PC，PS/2	1989年5月	PC-DOS 2.0以上，メモリ512Kbytes
		5.0	ボーランド	FM-R PanacomM	1989年12月	MS-DOS 2.1以上，メモリ640Kbytes
◎	Turbo-Debugger &Tools	2.0	ボーランド	PC-9800	1990年7月	MS-DOS 2.1以上，メモリ640Kbytes
		2.0	ボーランド	IBM PC，PS/2	1990年5月	PC-DOS 2.0以上，メモリ384Kbytes
	gawk	2.11.1	GNU	UNIX，MS-DOS	1989年11月	日本語対応のjgawkは，2.11.1をベースにしてあり，バージョンは＋1.9
	bison	1.03	GNU	UNIX，MS-DOS	1989年8月	
	emacs	18.55	GNU	UNIX	1989年8月	
	gcc	1.37.1	GNU	UNIX	1990年2月	
	g++	1.37.1	GNU	UNIX	1990年3月	
	make	3.58	GNU	UNIX，MS-DOS	1990年2月	
◎	scheme	7.0	GNU	UNIX	1990年6月	
	smalltalk	1.1	GNU	UNIX	1990年5月	
○	file utilites	1.3	GNU	UNIX	1990年6月	内容は，cat，chmod，cmp，cp，create，dd，dir，du，head，install，ln，ls，mkdir，mkfifo，mv，rm，rmdir，tac，tail，vdir

表の左端はチェック欄で，最近大きな変化（メジャー番号の変更）があった場合に◎が，小さな変化（マイナー番号の変更）があった場合には○が付いている．対応機種については，原則としてシリーズ名で表示しているが，特定の機種のみに対応しているものについては具体的に表示している．リリース欄にある日付は，掲載した最新バージョンの発売（出荷）時期である．データは，7月上旬の調査によるものであり，それ以降にバージョンアップが行なわれた可能性がある．

アンケート・ハガキの記入方法

アンケート・ハガキは，本誌Super EXPRESS(P. 33)の前にとじ込んであります．アンケート・ハガキの表面の空白部分には，スーパーアスキーへのご意見・ご感想などをお書きください．イラストなども歓迎いたします．書いていただいた文章などは本誌読者からのお便りコーナー欄に掲載させていただくことがあります．

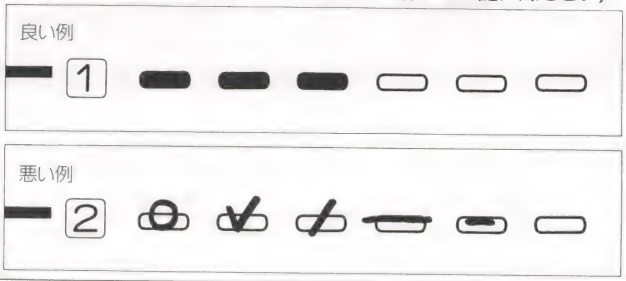
裏面のマークシート欄は，以下にあげる質問について，該当する番号の枠内を記入例のようにHBなどの濃い鉛筆でマークしてください．ハガキには，質問事項が書かれていませんので，番号をよくお確かめの上，ご記入くださるようにお願いします．マークシート以外の部分，およびハガキの表面は鉛筆以外のペンで記入していただいてかまいません．

皆様からお送りいただいたアンケート・ハガキは，スーパーアスキーの内容をより充実したものにするための参考にさせていただきます．ぜひご協力ください．「読者プレゼント希望」の欄には，P. 189に掲載したプレゼントの中から希望するプレゼントの番号および商品名をお書きください(希望する物がない場合は空白にしておいてください)．

毎月10日までに到着したハガキの中より，抽選で50名の方に図書券500円分をお贈りいたします(ただし，プレゼントの抽選を優先して行ないます)．なお，当選者の発表は発送をもって代えさせていただきます．また，読者からのお便りコーナー欄に採用させていただいた方には，図書券やオリジナル・グッズなどをさしあげます．

ハガキは切り取り線からきれいに切り離し，折り曲げたり汚したりしないようにご注意ください．

●マークシート記入例(HBなどの濃い鉛筆でご記入ください)



●マークシート部分は以下のアンケート項目についてお答えください．

1 次の中から面白かった記事の番号を挙げてください(5つまで)．

①表紙
②Super EXPRESS
③イベントカレンダー
④特集・オペレーティングシステム／本編
⑤特集・オペレーティングシステム／debate
⑥INSIDE SOFTWARE　OS/2　Ver. 2.0
⑦TEST LAB.／TOOLBOOK
⑧　〃　／SPARC SLC
⑨　〃　／QMS ColorScript100
⑩ソフトウェア＝ハードウェア＋ソフトウェア
⑪Virtual Reality
⑫最先端システム探検隊
⑬Astro Guide '90
⑭OS/2アーキテクチャ入門
⑮プロフェッショナルワークステーションとしてのUNIX入門
⑯プログラミング・フォーラム／386 de MS-DOS
⑰　〃　／ウィンドウシステムのプログラミングパラダイム
⑱　〃　／システムプログラム
⑲EPnews
⑳MIPS／パソコン1024夜
㉑　〃　／プログラマのための数学的世界観入門
㉒　〃　／ウサギと月と，そして太陽と
㉓　〃　／Fat Bits
㉔　〃　／Book Shelf
㉕　〃　／最新ソフトバージョンリスト
㉖読者のコーナー

2 つまらなかった記事の番号を挙げてください(5つまで)．

上記の項目の中からお選びください．

3 本誌以外にご購読のパソコン雑誌は何ですか(いくつでも)．

①日経パソコン②日経バイト③日経コンピュータ④日経エレクトロニクス⑤The BASIC⑥パソコン・マガジン⑦THE COMPUTER⑧インターフェース⑨トランジスタ技術⑩月刊ASCII⑪UNIX MAGAZINE⑫NETWORKER⑬その他

4 本誌の購読についてお聞かせください．

①ほとんど毎号買う②興味ある記事があれば買う③あまり買う予定はない

5 コンピュータの使用状況をお知らせください(いくつでも)．

①自分で所有しているコンピュータを使用
②家族や友人のコンピュータを借りて使用
③勤務先や学校のコンピュータを使用
④まったく使っていない

6 コンピュータを使用する目的は何ですか(いくつでも)．

①仕事②学業③趣味・遊び

7 コンピュータは主にどのようなことにお使いですか(いくつでも)．

①ゲーム②ワープロ③データベース④雑誌などのプログラムリストの入力⑤自分でプログラムを作成⑥パソコン通信⑦コンピュータ・ミュージック⑧コンピュータ・グラフィックス⑨表計算・作表⑩技術計算⑪コンピュータによる教育・学習⑫パソコン自体の学習⑬その他

8 次に購入したいと思っているものは何ですか(1つだけ)．

①コンピュータ本体②ハードディスク③拡張メモリ④プリンタ⑤スキャナ⑥モデムおよび通信ソフト⑦CG関連のソフトまたはハード⑧コンピュータ・ミュージック関連のソフトまたはハード⑨表計算・データベースソフト⑩ワープロ・エディタ⑪ゲームソフト⑫CやBASICなどの言語処理系⑬ユーティリティソフト⑭その他

9 年齢

①10歳未満②10～14歳③15～19歳④20～24歳⑤25～29歳⑥30～34歳⑦35～39歳⑧40～44歳⑨45～49歳⑩50歳以上

10 性別

①男性②女性

11 職種(1つだけ)

①小学生②中学生③高校生④短大・専門学校・予備校生⑤大学・大学院生⑥技能・労務職⑦営業・販売職⑧専門・技術職⑨事務職⑩管理職⑪自営業⑫その他⑬無職

Editor's Note

広がるウィンドウ環境

▽先月号のINSIDE SOFTWAREでWindows Ver.3.0を取り上げたのに引き続き，今月号のTEST LAB.では，その上で動作するハイパーテキストToolBookを紹介した．山崎俊一さんがFat Bitsでも触れられているように，米国におけるWindows Ver.3.0の動向は，大変好調のようだ．

▽ところで，パーソナルコンピュータの世界に限らず，最近はウィンドウ環境が広まりつつある．コマンド・インターフェイスからメニュー・インターフェイスへの移行は，一見，時代の流れであり，疑問の余地はないように思える．また，この流れは，CUIからGUIへの移行と相俟って，ユーザーに容易な操作と豊かな表現力を与えてくれる．

▽しかし，一方でこの移行のために必要となるコンピュータ資源の大きさは馬鹿にできない．編集部では当初Windows Ver.3.0の評価にクロック周波数10メガの80286マシンを利用していたのだが，ToolBookの評価には，速度的に厳しいものを感じていた．その後25メガの80486マシンを利用するようになり，ようやく十分に快適な環境となった．米国では，Asymetrix社がToolBookのチューンアップを引き続き行なうとアナウンスしているが，現状では，少なくとも16メガの80386マシン程度は用意したいところだ．

▽かつて，コンピュータでは日本語を利用できないのが当然の時代があった．それが，現在では，日常的に利用できるようになっている．マン・マシン・インターフェイスの変化は，コンピュータの位置付けを変えてしまうほどだ．が，一方で従来の価値観からすると想像を超える資源を必要とすることも多い．そこまでする必要があるのかという疑問も，必ず発生する．本誌では，そうしたバランス感覚も大切にして，この大きな変化を紹介し続けていきたいと考えている．

（土田米一）

編集後記

■40万円で何を買うか．編集部でISDNを導入することになったので，自宅にもISDNを引こうか，それとも新しいSVHSのビデオと衛星放送のセットを買うか．386の33MHzのマザーボードとSVGAボードも欲しい．しかし，本当に欲しいものは金では買えない．（土屋）

■ちょっと前までは，指輪などというものをはめるとキーボードを打つときに邪魔になるのではないかと考えていたのだが，どうもそういうことはないようである．しかしクラッチレバーにはカチカチと当たってくれる．こちらは，まるで予想していなかっただけになかなか不気味．（五十嵐）

■夏が来た．編集部Ｉ氏の結婚騒動の余韻も醒めやらぬとある日，NUWV同期のAから電話があった．「俺今度結婚することになったから」……．ををををを！ なんてこったい．どうしてみんなそんなに「あちら側」に行き急ぐんだあ，と不条理な独り言でも呟いてみる．今年も暑い暑い夏が来た．（JIN）

■ふとしたことから，20年前の男性週刊誌（「話のネタ本」とかいう，すごくストレートな名前です）を見る機会を得た．巻頭カラーは20歳のピチピチギャル（ちなみに，露出度は現在のそれに比べるときわめて控えめです）．でもこのモデルさん，今はもう40歳なんですよね．うーむ．（小川）

■最近，胃が痛ひ．かつては「金もストレスもためられない奴」，「厚顔無恥」，「偉そう」など，ありとあらゆる“賞賛の言葉”を浴びて，涼しい顔をしていたのが嘘のようだ．とはいえ，「ようやく大人になったんだよ」などと言われても，嬉しいはずもないのであった．（ToSHI）

■６月末にDigital World取材でロサンゼルスへ行ってきた．ロスの前にサンフランシスコでも取材したのだが，東京を発った日がセ氏30度の猛暑，シスコは14度でヒンヤリ，そしてロスでは史上最高の40度を記録した．人間てかなり順応性に富んでいるのだなー（これ実感！）．（沼尾）

■去年，作ったウェットスーツがとてもきつい．太ったことは明らかなのだが，怖くて体重計には乗ることができない．で，減量を決意したのだが，夏のビールとO'ZACK（小室哲哉がCMしているポテトチップ）の誘惑には勝てず，ついつい手が…．（コンビニの明かりが恨めしいこばやし）

■最下位を走る自国の選手を「我が偉大な祖国の後を追う者は誰もいない」と報道したという共産圏の逸話がある．確かに嘘は言ってはいないが，マスコミ情報の真実とは何か考えさせられる話だ．聞けば創刊号は各地で品切れ続出だったとか．偉大なスーパーアスキーの後を追うものは…．（聖）

■私はハイテクものにとても弱い．この間，念願のCDラジカセを買った（会社の人にも自慢してやった！）．好きなCDも買い，部屋で１曲，２曲．「音楽のある生活っていいな」と独り言を言い，６曲目ぐらいで，“B面にしなくっちゃ”とCDラジカセに向かう私でした．（面野）

■先日，アウトドア雑誌の方から電話取材を受け，「いかにも第１次産業の匂いのする名前ですね」と褒められた（？）．「名前など人を一意に識別するための記号だ」とは我が父親の言だが，その割に妙に名字とフィットする．「農水省に勤めたら」とからかわれた昔を，ふと思い出す．（土田）

月刊スーパーアスキー 10月号は9月18日（火曜日）発売！

特集 中規模ネットワーク構築計画
機種ごとフロアごとのLANをつないでみる

PCやWSのLANは，ほぼ形が見えてきている．プリンタやファイルを共有するだけならこのレベルでもよいが，中規模の企業でオンライン業務の道具として使ったり，コミュニケーションの道具として使ったりするためには，ルータやゲートウェイを導入して，複数のLANを基盤とした中規模ネットワークを構築せねばならないのである．

●発行人 塚本慶一郎 ●編集長 土田米一 ●AD （株）ポパイ 溝端ひろみ ●副編集長 土屋信明 ●編集 五十嵐久和／永瀬 仁／小川誉久／渡邉利和／沼尾禮子／小林章彦／水沼裕子 ●制作 （株）ポパイ 面野かおる／江面信幸／高木由美／吉積里枝／山口美和子 ●進行 聖 ●撮影 内野 進／田中健二 ●校正 秋山真実 ●協力 上野和明／遠藤孝信／海江田一詩／高木淳彦／谷口 崇／柳沢俊行／菅野康一 ●広告 佐藤敏明／秋山伴道 ●営業 渡辺昌之／鷲山直己 ●資材管理 勝又俊永 ●印刷 凸版印刷（株） ●各記事ごとの著作者名は該当する記事中に表示しました．

編集・著作 **株式会社アスキー**

〒107-24 東京都港区南青山6-11-1 スリーエフ南青山ビル
Tel（03）486-7111（大代表）
情報（内容のお問い合わせ）Tel（03）498-0299（ダイヤルイン）
雑誌営業部Tel（03）486-1977（ダイヤルイン） 1719009

ロングセラー

1971年の創刊以来、
先端技術者の情報源として
愛読されてきた「日経エレクトロニクス」。
いまや読者数は約5万人。
米国主要電子技術情報誌との全面提携や、
国内外の第一線技術者からの寄稿、そして
専門記者による直接取材を通じて、
仕事に直結する具体的なヒントを
毎号お届けしています。

ロングセラーの秘密は
情報の密度です。

最近掲載してご好評をいただいた記事です。

特集	カラー液晶搭載パソコン、3年後に普及へ
特集	分散コンピューティング環境の最前線
NEレポート	ノート型やカラーTFT液晶パネル搭載など、パソコンの新製品が相次ぐ
解説・産業	2つのUnix 業界標準賭けての技術競争は90年代前半へ
両相・逆相	日本人よ、コンピュータを恐れるな
解説・技術	3.5インチ光磁気ディスク、フロッピー代替から用途探る
論文	動画像の高能率符号化チップ・セットを開発
解説・技術	高速SRAM、特定マイクロプロセサに特化する方向へ
NEセミナ	ディスク・バックアップ・システムの最適設計

あなたもいますぐ

●個人の年間予約購読、ご自宅郵送制
●隔週月曜日発行、A4変型判、毎号約300ページ
●税込み料金：1年間（27冊）15,800円（本体 15,340円）
3年間（81冊）31,600円（本体 30,680円）
●料金のお支払い（一括払い）は、送本開始後、別便にてご案内いたします。

書店ではお求めになれません。

電子・情報技術の総合誌
日経エレクトロニクス
NIKKEI ELECTRONICS

発行 日経BP社

24時間体制で受け付けています。

お申し込みは 裏面のハガキかお電話で！ ☎(03)5696-6000

資料請求 No.00A

先端技術者の

情報源。

「日経エレクトロニクス」は、広がるエレクトロニクス分野のすべての情報を、早く確実に、わかりやすくお届けする情報誌。テクノロジや産業構造面で、大きな変貌が予想される1990年代の電子・情報技術分野。その最新動向を、技術・製品・産業の3つの視点から解き明かします。あなたと電子を結ぶホットライン＝「日経エレクトロニクス」をいますぐご購読ください。

技術 / 情報 / 製品 / 産業

技術

電子技術の"新しい流れ"をキャッチアップ。

めまぐるしく進化する技術情報をいち早くキャッチすることが、エレクトロニクス技術者にとって、今後ますます重要となるでしょう。「日経エレクトロニクス」は、エレクトロニクス分野における、さまざまな技術の最前線をクローズアップ。あなたの明日の仕事に役立つ情報を、どこよりも早くお見せします。

最近ご好評いただいた記事は…

- A-D変換器、ディジタル・チップにのる
- 知的財産権、「ひらめき」を売る時代へ
- 偏向板不要で明るいディスプレイ実現、高分子分散液晶が台風の目に
- 人工現実の世界でシステム・キッチンのレイアウトや使い勝手を検証する
- 日本のディジタル自動車・携帯電話、標準方式の大枠決定
- 携帯型パソコンの将来像を探る
- 64MビットDRAMの試作が始まる
- 携帯性、操作性に優れた肩掛けコンピュータを医学的見地から提案する
- 脱フロン時代の洗浄技術

製品

「何をつくるか」の詳細な情報とヒントを提供。

いまやエレクトロニクス技術者に求められているのは、量産化などの「いかにつくるか」ではなく、「何をつくるか」という柔軟な発想です。「日経エレクトロニクス」は、製品企画のひとつひとつに焦点を合わせ、企画現場やその裏側をスクープ。製品企画や設計に直接役立つ詳細な情報とヒントをあなたにお届けします。

最近ご好評いただいた記事は…

- DRAMに専用化の道を開いた画像用256Kデュアル・ポート
- 生き残りを図る個別半導体メーカ、外資系も市場参入進める
- i486に直結できる128Kバイトのキャッシュ・モジュールの出荷を開始
- 家庭用をねらったハイビジョン・カメラを試作
- Macintoshの最上位機発売、エンジニアリング用途狙う
- ROM DOS登場でMS-DOSの用途拡大へ

産業

電子産業の幅広いトレンドをカバー。

エレクトロニクス技術者にとって、ユーザや市場のニーズと動きを知らなければ、「何をつくるか」の答えを出すことはできません。こうした観点から、「日経エレクトロニクス」は、電子産業の市場、経済動向を積極的にカバー。エンジニア自身の立場と視点にこだわり、確実に使える情報を提供しています。

最近ご好評いただいた記事は…

- 鉄鋼大手の半導体ビジネス、いまだ腹を括れず
- 光産業、1989年度は20%成長し2兆4983億円
- 携帯電話 店頭売り切りの「民生市場」へ向け胎動
- シリコン・バレーに未来はあるか
- 半導体、付加価値を高めつつ「ギガ」を目指す
- 米国が知的財産権保護の姿勢強める

隔週月曜日発行

電子・情報技術の総合誌

日経エレクトロニクス
NIKKEI ELECTRONICS

発行 日経BP社

書店ではお求めになれません。
購読のお申し込みは、左のハガキかお電話で。
お急ぎの方はお電話でいますぐ。
☎(03)5696-6000

日経エレクトロニクス NIKKEI ELECTRONICS 新規購読申込書

お名前		フリガナ		年齢	歳
ご自宅住所	(フリガナ) □□□-□□				
ご自宅電話	( ) －	会社電話	( ) －		
お勤め先名		所属部課名	内線TEL( )		
ご卒業校	専攻学科		卒業年度 19 年		

A・Bどちらか希望される方に○をお付け下さい。

購読料金(税込み料金)	A. 1年(27冊)15,800円(本体15,340円) B. 3年(81冊)31,600円(本体30,680円)	お支払い方法	A. 郵便局窓口からの現金振込 B. 銀行預金口座からの自動引落し

あてはまる番号に○をお付け下さい。

職種	1.研究・開発 2.設計 3.生産技術・管理 4.購買 5.営業 6.保守サービス 7.経営部門 8.教育・その他
役職	1.役員 2.管理職クラス 3.社員・職員
専門技術分野	1.材料・物性・部品・デバイス 2.回路 3.コンピュータハードウエア技術 4.ソフトウエア技術 5.通信 6.非電気・電子技術 7.非技術系
関連製品分野	1.材料・部品・デバイス 2.汎用コンピュータ 3.パソコン・関連システム 4.ソフトウエア 5.周辺端末装置 6.計測器 7.通信器・通信システム 8.その他の産業用機器・装置 9.民生用機器・装置 10.データ処理、特許など

※途中解約時のご精算は(お届け済冊数×税込み1冊価格1,030円)でお願いします。

Sアスキー02-42-0043

料金受取人払

葛西局承認

542

差出有効期間
1992年3月31日
まで(切手不要)

郵便はがき

134-00

(受取人)
東京・葛西郵便局私書箱20号
日経BP社
読者サービス・センターNE係行

Fuji Software Inc.

FSI Original Products

富士ソフトウエアではMS OS/2プレゼンテーションマネジャ及びUNIX用アプリケーション、フォントの開発・販売を致しております。販売形態はOEM(ソース及びオブジェクト)、パッケージ販売などご要望に応じます。また以下のソフトを基にご希望のシステムの開発も承ります。

OS/2

PM対応日本語FEP Felix/2(仮称)

●標準日本語環境であるKKC(Kana To Kanji Conversion Module)に準拠、本体はDLL(Dynamic Link Library)として提供します。

●ワープロ、データベースなどのそれぞれのウインドウに対して、きめ細かな環境設定が可能です。

●プレゼンテーションマネジャの機能を生かしたビジュアルなインタフェースを採用し、キーのカスタマイズ、サイクリック変換などのユーザーフレンドリーな操作性を実現します。

PM対応 Desktop Accessory Pocket Works(仮称)

●Pocket Worksは文房具感覚で使える、便利なアクセサリ集です。

【ワープロ】簡単な1ページ程度の書類なら、文章、表を同時に作成できます。

【電卓】演算は10進、16進ともに16桁まで演算可能です。ロールペーパー機能で計算過程が一目でわかり、演算結果のカット&ペーストも可能です。

【時計】アナログとデジタルで二つの時刻を設定可能です。またアラーム機能も装備しています。

【タイマー】アプリケーション実行中でも、アラーム音または画面フラッシュでタイムアップを知らせます。

【メモ帳】思いついたときにいつでもメモでき、10ページ(12行24桁/ページ)まで入力可能です。

【電話帳】氏名、住所、電話番号などを200件登録できます。ソート、検索も可能です。

GPIBデバイスドライバ

●MS OS/2用のバイモーダルなデバイスドライバです。非同期割り込み及び非同期入出力をサポートしていますので、MS-DOS環境では難しかったサービスリクエストの受け付けからシリアルポール、データ転送に至る一連のプログラムをマルチタスク対応で書くことができます。また他のデバイスドライバと並行動作ができるように、割り込み処理ルーチンはコンパクトに作られています。OS/2のAPIにスタイルを合わせたDLLも併せて提供致します。

イベントレコーダー(仮称)

●イベントの監視などのシステム情報の収集を行い、システムの保守、監視を容易にします。

オートパイロット(仮称)

●PM上で行うオペレーションを記録し、記録データの一部を加工することにより、バッチ処理やオートデモンストレーションを実行します。

UNIX

かな漢字変換ライブラリ SACK

●日本語処理機能をサブルーチンとして提供致します。C言語で記述していますので移植性に優れています。また端末データベースを使用しているので、特定の端末に依存しません。

辞書

●5万語及び3万語辞書を用意しております。テキスト形式なので辞書のメンテナンス、他OSへのコンバートが容易に行えます。

毛筆行書体メタフォント

●UNIX用毛筆スケーラブルフォントです。必要な大きさのTEX用フォントを生成することができます。

FONT

明朝体(開発中)
ゴシック体(開発中)
毛筆行書体(JIS第1・第2水準)
毛筆楷書体(JIS第1・第2水準)
イラストフォント(300種)

●アウトライン、ビットマップ、各種データフォーマットに対応致します。

OS/2 and UNIX

※MS-DOSは米国Microsoft社の登録商標です。
※MS OS/2は米国Microsoft社の商標です。
※UNIXは米国AT&Tが開発し、ライセンスしています。

●開発・販売
富士ソフトウエア株式会社
〒108 東京都港区三田3-4-3 三田第一長岡ビル
営業部
TEL03-455-7731 FAX03-453-4643

Cでのプログラミングでお悩みの

C言語に強力なデータベース機能を提供。

AccSys（アクシス）

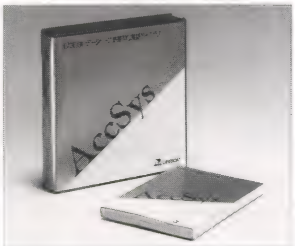
【対応C言語】
▶Lattice C Ver.4.1
▶Microsoft C Ver.5.1

価格 ¥98,000

New AccSysライブラリソース付
価格 ¥198,000

(dBASE互換) データベース処理用C言語ライブラリ

再販ライセンス付

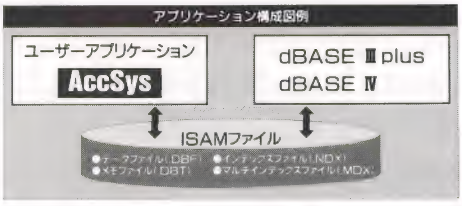


AccSysは、C言語プログラムで、データベース処理を実現するための、C言語用関数ライブラリです。これまでC言語での業務用アプリケーションの開発において最も労力を必要とし、ネックになっていたISAMファイルの構築や、リレーショナルアクセス等のデータベース処理が、AccSysの登場によりC言語で簡単に実現出来るようになりました。また、AccSys関数によって構築されるISAMファイルは、既存のRDBシステムdBASEⅢplusやdBASEⅣと互換性を持っているため、AccSysを使用する事によって、dBASEシステムと互換性のある業務用アプリケーションプログラムの開発が行えます。
C言語で独自のデータベース操作を行う事により、より高速で柔軟な、使用目的に合った業務用アプリケーション処理が可能になります。

【対応機種】 ▶PC-9801シリーズ ▶FMRシリーズ ▶Panacom Mシリーズ ▶J-3100シリーズ ▶IBM PS/55シリーズ ▶PC/AT ▶各社AX仕様マシン

AccSysの特長

●LRU(Least Recentry Used)メモリ・マネージメント・テクニックによって、より少ないメモリ容量で、より速いファイルI O操作を実現。
●dBASEⅢplusで作成された、既存データファイル及びインデックスの読み出し、更新及び追加か可能。
●dBASEⅢplus、dBASEⅣ互換なデータファイル、インデックスファイル、メモファイルの各ファイル操作をサポート。
●1つのファイルで、47個のインデックスか扱える、マルチインデックスMDXファイル操作を追加サポート。
●MDXインデックスキー操作は昇順・降順の両方をサポート。
●インデックスキーの重複か可能。
●データレコードの特定フィールドによるソーティングが可能。
●最大15個のファイルを同時オープン可能。
●データファイルの仕様。▶レコードの最大数：10億 ▶1レコードの最大長：4000バイト ▶1レコードに含まれる最大フィールド数：255フィールド ▶1フィールド(文字型)の最大長：254バイト▶フィールドの種類：文字型、数値型、日付型、論理型、メモ型



C言語でPC-9801を自在に操作。

C-TOOL/98

【対応C言語】
▶Lattice C Ver.4.1
▶Microsoft C Ver.5.1

ノーマルモード専用版
価格 ¥32,000

New ノーマル・ハイレゾ共用版
価格 ¥38,000

PC-9801対応C言語多目的ライブラリ

再販ライセンス付

完全ソース付
豊富なサンプルソース付

C-TOOL/98は、C言語用に、LIFEBOATが企画・開発した PC-9801 シリーズ用多目的ツール・ライブラリです。NEC PC-9801シリーズ上で、グラフィックスやマウス、画面カーソルやファンクションキーの制御、プリンター制御やサウンドボート制御などのハードウェア制御、さらに10進演算（BCD演算）等を含めて、C言語プログラミングで必要となる機能を、幅広くフレキシブルにサポートします。

【対応機種】 ▶PC-9801E／F／M／VM／VX／U／UX／RA／RX／RL／XL／XL²(但し、RL、XL、XL²のハイレゾモードでは、ハイレソ対応版でのみ使用可能)
●マウス（バス・マウス）(マウス関数を使用する場合)
●PC-9801-26Kサウンドボード(サウンド関数を使用する場合)

C-TOOL/98の特長

■BCD演算関数
16桁までの有効数字からなる浮動小数点10進数を扱う関数です。桁落ちや、誤差があっては困るビシネス処理のプログラミングに適しています。

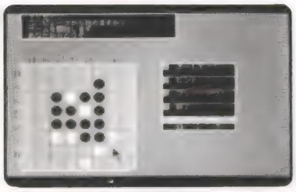
■プリンタ制御関数
プリンターをより細かく制御するための関数です。漢字の間の余分なスペースを開くことや、タブのサポートなどを解決できます。

■グラフィック関数
PC-9801シリーズのグラフLIOをサポートする基本グラフィックス関数はもちろん、ポリゴンか簡単に扱える拡張グラフィック関数も追加されていますので、BASICよりも機能か高いグラフィックス・プログラムか簡単に作れます。

■サウンド関数
PC-9801シリーズのサウンドボート（PC-9801-26K）を、C言語から簡単にコントロールできるようになります。
N88-BASIC(86)で提供されているサウント拡張命令のサブセットで、BASICとの互換性を保つように作られています。MML（ミュージック・マクロランゲージ）での作曲や、サウント関数による音楽演奏を楽しめます。

■マウス関数
マウストライバーを利用するマウス関数て、マウスインターフェース・プログラムか簡単に作れます。本製品附属のマウストライバーたけてなく、MS-DOSやマイクロソフト社Multiplanに附属されているマウストライバーにも対応しています。

関数名	機能
g_mktile	8色モード用のタイルパターンを作成する
g_polygon	多角形の中を塗りつぶす
g_polyfill	多角形の中を塗り直す
g_polyline	多角形の頂点を線で結ぶ
g_print	日本語ストリングを表示する
g_load	VRAMイメージ(96K)をディスク・ファイルからロードする
g_save	VRAM領域(96K)をディスク・ファイルへセーブする



■ハイレゾモード追加関数
PC-9801 RL、XL、XL²のハイレソリューションモートでは、さらに下の機能か使用てきます。

関数名	機能
t_cls	テキスト画面をクリアする
ishireso	ハイレソかとうかチェックする
g_wmode	直接GVRAMアクセス時の色を指定する
t_start	テキスト画面の表示を開始する
t_stop	テキスト画面の表示を停止する
t_csron	カーソルの表示を開始する
t_csroff	カーソルの表示を停止する
t_crtmode	テキスト画面のモートを設定する
t_crtsen	テキスト画面のモートを得る
t_ctrlf	CTRL＋f･1～f･15のソフト化 解除

■BASICライク関数
カーソル位置のロケートや、文字色の設定、ファンクションキーの設定等、N88-BASICのコマンドの一部をC言語でサポートします。

関数名	機能
t_locate	テキスト画面のカーソル位置を設定する
t_color	テキスト画面の文字の色を設定する
t_cls	テキスト画面をクリアする
t_fget	ファンクションキーの文字列を入手する
t_fput	ファンクションキーに文字列を定義する
beep	内蔵スピーカを鳴らす
inputs	プロンフット出力後に文字列を入力する
t_tojis	シフトJISコートをJISコードに変換する
t_tosjis	JISコードをシフトJISコードに変換する
t_delay	約18m秒単位で止まる

C言語に本格的な数学演

ピタゴラス

科学技術計算 & 統計処理用C言語

かって科学技術計算といえばFORTRANと決まっていましたが、最近ではコード生成効率の良いC言語がよく使われます。しかしFORTRANの世界では標準ルーチンと言っていいScientific Subroutine Package (SSP)がC言語にはなかったので、Cによる本格的科学技術計算は不可能でした。ところが、ピタゴラス(SSP plus)の登場によって、今や、それが可能になりました。
ピタゴラスには、数値解析に不可欠なルジャンドル多項式、チエビシェフ多項式をはじめ、ガンマ関数、ベータ関数、各種分布関数の他に、行列計算、複素数計算、FFTなど幅広い用途に応えるサブルーチンが集められています。

【対応機種】 ▶PC-9801シリーズ ▶FMRシリーズ ▶Panacom Mシリーズ ▶J-3100シリーズ ▶PS/55シリーズ ▶PC/AT ▶各社AX仕様機

C言語でカナ漢字変換処

EG日本語

日本語入力 & 変換処理用C言語ラ

多くのアプリケーションプログラムは、入力フィールドごとに、日本語を入れるか、数字またはアルファベットを入れるかが決まっています。かな漢字変換フロントエンドプロセッサがあれば、ユーザーにCTRL＋XFERを押してもらって、入力モードを切り替えればいいのですが、できれば、プログラムで自動的に日本語入力モードに切り替えたいものです。このとき、入力モードを自動的に切り替えるには、従来は、かな漢字変換フロントエンドプロセッサのプログラムインターフェースをフロントエンドのデバイスドライバを解析したり、メーカー公表の資料を参考にして利用していました。しかし、これらのインターフェースは、メーカーやバージョンの相違により変わってしまいます。他社製のフロントエンドに依存せず、ユーザーに安定した製品を提供するためには、変換機能ごとアプリケーションプログラムに組み込むことです。このEG日本語は、ライブラリ形式で、日本語変換機能を実現しているので、フィールドごとの入力モードの切り替えを自由にプログラムから制御できます。EG日本語によって、ユーザーフレンドリなアプリケーションを容易に実現できます。

■日本語編集機能
このライブラリは日本語テキストの編集機能を提供するものであり、複雑な日本語処理もこの機能を使用することにより簡単に実現することができます。

日本語編集ライブラリ関数一覧

SMSetText	編集テキストの設定
SMGetText	編集テキストの取り出し
SMIdle	カーソルのブリンク
SMActivate	編集テキスト定義
SMDeactivate	編集テキストの未定義
SMKey	文字の入力
SMGetCry	確定文字列の取り出し
SMCut	文字列のカット
SMCopy	文字列のコピー
SMPaste	文字列のペースト
SMDelete	文字列の削除
SMInsert	文字列の挿入
SMUpdate	テキストの書き直し
SMScroll	テキストのスクロール
SMSetSelect	カーソル(選択文字列)の設定
PrgInit	パラグラフの初期化
PrgNew	パラグラフ編集の開始
PrgEnd	パラグラフ編集の終了
PrgKey	パラグラフへの文字の入力
PrgIdle	カーソルのブリンク
PrgUpdate	パラグラフの領域の書き直し
PrgHeight	パラグラフの高さを得る

方々へ朗報!! We sell KNOW-HOW for Professional Programming

算サブルーチンを提供。

(SSP plus)

【対応C言語】
▶Lattice C Ver.4.1
▶Microsoft C Ver.5.1

価格 ¥98,000

ライブラリ　再販ライセンス付

ピタゴラス(SSP plus)の関数機能

▶指数／対数関数▶べき集計算▶誤差関数▶各種微分関数▶各種積分関数▶逆誤差関数▶各種ベッセル関数▶各種エアリ関数▶各種ケルビン関数▶ホイマン・ラムダ関数▶ワイアシュトラウスのP関数▶各種多項式▶各種分布関数▶各種疑似乱数及びRAND関数の種の変更▶最大値，最小値，平均値▶回帰直線・相関係数の算出▶スプライン補間▶最小自乗▶高速フーリエ変換▶複素数の各種演算▶各種行列計算▶コレスキー分解等▶約200個のサブルーチンを集録

●処理実行中のエラー・ハンドリング用サブルーチンも用意

理をFEPなしで実現。

【対応C言語】
▶Lattice C Ver.4.1
▶Microsoft C Ver.5.1

価格 ¥98,000

イブラリ　再販ライセンス付

【対応機種】
▶PC-9801シリーズ
(LT,ハイレゾを除く)

再販可能な辞書付き

開発元エルゴソフトにご連絡・登録いただくだけで、無料で辞書の再販が可能です。

■EG日本語 VS FEP比較表

	EG日本語	日本語FEP利用のプログラム
長所	プログラムによる制御が容易あらかじめFEPを組み込む必要なし。 ユーザーが別の製品(FEP)を購入する必要なし(ユーザーの負担軽減)。 日本語処理をプログラムする必要なし。	日本語処理をプログラムする必要なし。
短所	辞書を添付するので、製品ディスク使用量が増える。	FEPを指定し、ユーザーがそれに併せてFEPを用意する必要あり。 FEPのバージョンアップに合わせて、既存のプログラムの確認の必要あり。 他製品が別のFEPに対応している場合、FEPの組み込み直しが必要。

■日本語変換機能

アプリケーション・プログラムが日本語ワードプロセッサ等、日本語の入出力の良否が製品の良否に影響するような場合には、この日本語変換ライブラリを使用することにより、本格的な日本語処理が実現できます。

日本語変換ライブラリ関数一覧

EGCINIT	初期設定
EGCTXCVT	文章変換
EGCCLCVT	文節変換
EGCRMCVT	ローマ字変換
EGCJACVT	半角変換
EGCAJCVT	全角変換
EGCSJCVT	シフトJIS・JIS変換
EGCJSCVT	JIS・シフトJIS変換
EGCLEARN	同音語学習
EGCWDADD	ユーザー辞書登録
EGCWDDEL	ユーザー辞書削除
EGCFIND	辞書の検索
EGCNEXT	辞書の次検索
EGCREAD	辞書の読み出し
EGCCREATE	メイン辞書作成開始
EGCWRITE	メイン辞書作成書き出し
EGCEOF	メイン辞書作成終了
EGCNDIC	メイン辞書作成

C言語で計測機器や周辺機器を制御。

connect:GPIB

ライブラリソース付　価格 ¥38,000

GPIBデバイスドライバー & C言語インターフェース関数ライブラリ

【対応C言語】
▶Microsoft C Compiler Ver.5.1
▶Lattice C/DOS Compiler Ver.4.1

現在、計測機器や周辺機器には、IEEE-488標準デジタルインターフェース(GPIB)を持つ物が多いですが、GPIBインターフェースを持った機器の制御を、パソコン上から、しかもC言語のプログラムで、すぐに実現できます。

connect:GPIBは、NEC製PC-9801-29(K,N)標準GPIBボードに対応した、GPIBコントロール・デバイスドライバーとC言語のプログラムで、そのまま操作可能なインターフェース関数ライブラリーを含みます。ライブラリ部分は、完全ソースで提供されますので、インターフェース関数のユーザ側でのプログラム仕様に合わせたライブラリの書き換えも可能です。(アセンブラからは、int D1HのBIOSコールでアクセスできます。) MS-DOS上で、GPIBインターフェースを使用したアプリケーションの開発効率を向上できます。

■BIOSファンクション(int D1H)

FUNK.No.	名　称	機　能
0	CINIZ	コントローラのイニシャライス(IFC・RENの実行)
1	SET IFC	IFラインをアクティブにする
2	SET REN	RENラインを
3	RESET REN	RENラインを
4	SEND CMD	GPIB上にコマ
5	SET DELIMT	デリミタコー
6	SEND DATA	GPIB上にテー
7	RECEIVE DATA	トーカからの
8	EXECUTE SERIAL POLL	シリアルホー
9	SET SRQ	GPIB上にSRQ
10	EXECUTE PARALLEL POLL	ハラレルホー
11	SET PPR MODE	ハラレルホー
12	SET TIME OUT	GPIBのハンク
13	CHECK STR	現在保持して

■インターフェース関数ライブラリ(完全C言語ソース付)

No.	関数名	機　能
1	ib_chkinstall( )	ドライバインストールチェック
2	ib_inizctrl( )	コントローラのイニシャライス
3	ib_ifc( )	インターフェイスクリアの実行
4	ib_setren( )	セットREN
5	ib_resetren( )	リセットREN
6	ib_txcmd( )	コマントの送信
7	ib_setlt( )	リスナ・トーカの指定
8	ib_txdata( )	テータの送信
9	ib_rxdata( )	テータの受信
10	ib_setdelimilt( )	デリミタのセット
11	ib_getdelimit( )	現在セットされているデリミタ値を得る
12	ib_spoll( )	シリアルポール実行後、SRQ送信装置のSTBを得る
13	ib_srqout( )	スレーブモード時にSRQを送出し、ステータスバイトをセットする
14	ib_timeout( )	タイムアウト値の設定

【動作環境】▶MS-DOS Ver.3.1／3.3
▶PC-9801シリーズ(GPIBボード搭載可能機種)
▶GPIBボード(PC-9801-29K／N)

C言語でRS-232Cポートを自在に制御。

connect:232C

完全ソース付　価格 ¥38,000

標準・拡張RS-232Cポート制御用C言語ライブラリ

再販ライセンス付

【対応C言語】
▶Microsoft C Compiler Ver.5.1
▶Lattice C/DOS Compiler Ver.4.1

connect:232Cは、標準関数ではサポートしていない種々の機能を有する高機能RS-232C制御ライブラリです。μPD8251SIO調歩同期式全2重の動作モードの設定、ステータスの読み込み、受信バッファの変更などを簡単に行うことができます。通信速度は19200BPSをサポート、フロー制御もサポートしていますので、MNPモデムなどにも対応可能です。

また、拡張ボード(PC-9861K)の第2、第3回線もサポートしています。各種漢字・文字コードの送受信や、XMODEM，YMODEMプロトコルによる送受信も行えます。

connect: 232Cライブラリ関数一覧

■高水準通信関数(ccom.c)

c_linit	通信関数使用環境の初期化
c_lendi	通信関数使用環境の終了時処理
c_lopen	通信オープン
c_lcose	通信クローズ
c_lrecv	1文字受信
c_lsend	1文字送信
c_lsendch	1文字送信(送信完了まで待つ)
c_lsendstr	文字列の送信
c_lctrl	信号線・フロー制御
c_lstatus	通信関数状態の取得
c_lbreak	ブレーク信号送出
c_lwait	時間待ち関数
c_lreset	フロー制御解除

■低水準通信関数(機種依存部分)(ccom.c)

c_binit	RS232Cの割り込み初期化
c_bendi	RS232Cの終了時処理
c_dopen	RS232Cオープン
c_bclose	RS232Cクローズ
c_brecv	RS232C1文字受信
c_bsend	RS232C1文字送信
c_bctrl	RS232C信号線制御
c_bstat	RS232C状態の取得
c_bbreak	RS232Cブレーク信号送出
c_bwait	時間待ち関数

■漢字処理通信関数(jcom.c)

c_jpset	漢字環境の設定
c_jrecv	1文字受信(漢字処理)
c_jsend	1文字送信(漢字処理)
c_js2sj	JIS→シフトJIS変換
c_sj2js	シフトJIS→JIS変換

■端末関数(tmcom.c)

c_tinit	端末関数環境初期化
c_topen	通信オープン
c_tclose	通信クローズ
c_tget	1文字受信
c_tput	1文字送信
c_tlog	ログファイル指定
c_txrecv	XMODEM受信
c_txsend	XMODEM送信
c_conin	コンソール1文字入力
c_conout	コンソール1文字出力
c_beep	ブザー出力
c_csrpos	カーソル位置の取得
c_csrmov	カーソル移動
c_csron	カーソル表示
c_csroff	カーソル非表示
c_cls	画面クリア

■アセンブラC-インターフェース変数一覧

c_rsloop	送信待ちループから脱出する為のフラグ
c_rsxon	文字フロー制御の有効 無効
c_rshxon	ハードウェアフロー制御の有効・無効
c_rssxof	送信フロー制御状態の保持
c_rsrxof	受信フロー制御状態の保持
c_rbcnt	受信データバイト数
c_rbrdp	現在の受信リードポインタ
c_rbwrp	現在の割り込みリードポインタ
c_rb-3-4	受信バッファサイズの3 4
c_rb-1-4	受信バッファサイズの1 4
c_rbufo	受信バッファオフセット
c_rbufs	受信バッファセグメント
c_rbufl	受信バッファの最大長

【動作環境】▶MS-DOS Ver.3.1／3.3
▶PC-9801シリーズ　▶拡張RS-232Cボード(PC-9861K)(標準ポートのみでも利用可能)

■パソコン同志のデータの交換に!
■モデムの制御に!
■通信ソフトの作成に!
■ホストシステムの構築に!

サンプルとして
ターミナルソフトつき

テキストのアップロード・ダウンロードはもちろん、XMODEMプロトコル・オートログイン機能を持つ本格的な高機能ターミナルソフトのソースコードをサンプルとしてお付けしています。
そのままでも充分パソコン通信を楽しめますが、自分で独自の機能を加えて、更に高機能な自分だけのターミナルソフトに改良していけます。

【お求め方法】 当社製品は、パソコンソフト取扱い店でお求めいただけますが、弊社に直接ご注文の際は、The Programmer's INDEX (TEL:03-293-3887)までお問い合わせ下さい。 ※各製品の表示価格には、消費税は含まれておりません。

●Copyright Notice
* AccSysは、Copia International, Ltd.の商標です。
* C-TOOL/98は㈱LIFEBOATの商標です。
* ピタゴラス(SSP plus)は、㈱LIFEBOATと㈲京葉インフォメーションプロジェクト、AMS, Inc.の提携品です。
* EG日本語は、㈱LIFEBOATと㈱エルゴソフトの提携製品です。
* connect:232Cは、㈱LIFEBOATと㈱ブート・システムズ・ジャパンの提携製品です。
* connect:GPIBは、㈱LIFEBOATと㈱エイピイラボの提携製品です。
* Microsoft、MS-DOSは、Microsoft Corp.の登録商標です。
* dBASE III、III plus，IVは、米国Ashton Tate社の登録商標です
* その他、プログラム名、システム名、CPU名は一般に各メーカーの(登録)商標です
* 価格、仕様は予告なく変更する場合があります

株式会社 LIFEBOAT
〒101 東京都千代田区神田錦町3-6
PHONE:03-293-4711㈹ FAX:03-293-4710

技術者の夢を実現するための、新しいOS環境“MS OS/2”。そのすべてを学んでみませんか。『Microsoft University』はアプリケーション開発に携わる技術者のために初歩から高度な段階まで9つのコースを設置。いま知りたいことに的確にお応えします。新たにLANコース、Microsoft C入門、OS/2入門コースを設けました。

MS OS/2でつぎの時代をデザインしてください。

●問い合せは
富士ソフトウエア株式会社
MU教育センター
TEL.
03-452-7722(直通)
FAX.**03-798-0599**

MS OS/2 Programming Environment (PE)

MS OS/2アプリケーションソフトを作成する場合に必要となる基本的な事柄について学びます。特に、OS/2の機能およびプログラム開発環境について解説します。本コースは、OS/2に取り組む上での基本コースとなります。

■期間：4日間 ■定員：20名 ■受講料：12万円(税別)

MS OS/2 Applications Programming (AP)

MS OS/2のAPIを使用し、マルチタスク、メモリ管理、プロセス間通信、ダイナミックリンクなどの機能を有効に活用するためのプロテクトモード・アプリケーションの解説を行います。実習中心のコースです。

■期間：5日間 ■定員：20名 ■受講料：15万円(税別)

MS OS/2 Presentation Manager Programming Environment (PM/PE)

MS OS/2プレゼンテーション・マネージャ(PM)のアプリケーションを作成するための入門コースです。作成に必要な基本事項を説明するとともに、実際に簡単なサンプルプログラムの作成を行います。

■期間：5日間 ■定員：20名 ■受講料：15万円(税別)

MS OS/2 Presentation Manager Applications Programming (PM/AP)

MS OS/2プレゼンテーション・マネージャ(PM)のアプリケーションを作成するための上級コースです。プロジェクトの中心となる方、経験者を対象に、より詳しく開発の方法、ノウハウを説明します。

■期間：5日間 ■定員：20名 ■受講料：15万円(税別)

Microsoft Windows Programming Environment (Microsoft Win/PE)

MS-Windowsのアプリケーションソフトを作成する方のための入門コースです。Windowsプログラミングのポイントを重点的に解説し、実際にサンプルプログラムを作成しながら学んでいきます。実習中心のコースです。

■期間：5日間 ■定員：20名 ■受講料：15万円(税別)

MS OS/2 Device Drivers (DD)

MS OS/2デバイス・ドライバの機能と構造を詳しく解説します。バイモーダル、ROM-BIOS互換サポート、デバイスヘルパルーチンなども詳しく解説します。実際にデバイス・ドライバのサンプルを作成し、インストール、デバックまで行います。

■期間：4日間 ■定員：20名 ■受講料：15万円(税別)

Microsoft LAN Manager Programming Environment (Microsoft LAN/PE)

マイクロソフトLANマネージャ(LM)用アプリケーションを開発する設計者のための入門コースです。LMのアーキテクチャと特徴について説明します。また、基本的なプログラムを作成できるようになります。

■期間：4日 ■定員：20名 ■受講料：15万円(税別)

Microsoft C入門 (MS-C)

C言語の基本を解説します。C言語の特徴である、関数、制御構造、ポインタについて詳しく説明します。セミナでは、講義だけでなくプログラムを作成し、実際にその動作を確認していただくことにより、理解を深めます。他の処理系とは違うMS-C独自の規約なども説明します。

■期間：2日間 ■定員：20名 ■受講料：5万5千円(税別)

MS OS/2入門

MS OS/2の基礎知識、オペレーション、開発環境を理解する初心者のためのコースです。これからOS/2を導入し、アプリケーションを作成するための知識が習得できます。また、実習中心のアクティブな講習であり、OS/2の操作や簡単なプログラムの作成を体験できます。

■期間：2日間 ■定員：20名 ■受験料：8万円(税別)

次世代標準OSのすべてを学ぶMS OS/2セミナ

Microsoft University

〈オンサイトコース実施中〉
受講者が10名以上の場合は、開催日程、カリキュラム等、貴社のご要望に合わせてオンサイトコースを実施いたします。
日本電気株式会社殿の御協力によりPC-9800シリーズ対応コースも好評開催中!!

Microsoft®
マイクロソフト株式会社
〒160 東京都新宿区西新宿7-5-25 Kビルディング

FSI 富士ソフトウエア株式会社
MU教育センター
〒108 東京都港区三田3-4-3 三田第一長岡ビル

日本語 フルスクリーン エディター　ブリーフ

Brief 日本語

■Full Screen Editor

好評発売中

IBM PCのトップセールスに輝くエディターが日本で4機種対応から5機種対応へ、さらに広がったBriefの世界

長所は広い交友関係です。

■NEC
PC9801 E（漢字ROMが必要）
PC9801 F/M/U/UV/VF/VM/VX/UX/LV/CV/RA/RX/LS/LX/ES/EX/RS
PC9801 XL/XL2/RL（ノーマルモードのみ）

■富士通
FMR 50 60 70
FM TOWNS

■IBM
PS/55 S/T/V/Z
5550 MPシリーズ

■東芝
J-3100 SS SL GT GX SGT
J-3300 30 50
（モノクロ・モードのみ）

■AX協議会
AXシリーズパソコン

革命的マルチマシン・エディター

一本のBriefを、AX協議会・東芝・NEC・IBM・富士通の5メーカー8ラインの機種にインストールできます。いずれの機種でも日本語DOSで縦横無尽の威力を発揮します。

オピニオンリーダー誌のこの評価

- プログラミング向きのエディターを探している人は、もう探す必要はない。
  Jerry Pournelle（BYTE）
- Briefはプログラマーに最適なだけでなく創造力をかきたてる何物かがある。
  Computer World
- ほとんど完ぺきにカスタマイズできるだけの拡張性がある。
  Computer language
- まるで夢のようなアンドゥだ—おもわず間違うのを期待してしまいそうだ。
  The C Usere Journal

●強力無比―マクロ言語

マクロは、極めて強力な表現能力と柔軟性を持った、完成度の高い構造化言語です。日本システムでは販売後のBriefにも新しいマクロを提供、利用者間のマクロの交流を図り、編集環境のグレードアップを支援し続けます。

●300回ものアンドゥ

Briefでは各ファイルごとに300までの操作を元に戻すことができますので、やり直しの面倒を気にせず、安心して編集できます。削除や置換、別のファイルの挿入など、ほとんどの編集結果をひとつの操作で元に戻せます。

●55ウィンドウ、無限のファイル数

ウィンドウは縦横自由に、55まで開くことができます。また、同時に編集できるファイルの数や大きさには制限がありません。ひとつのファイルの複数の部分を、それぞれ別のウィンドウ上で同時に編集することも可能です。

●プログラミング

◆キー操作ひとつでコンパイルが可能。またプログラムにエラーがあれば自動的にその箇所へ移動。◆メモリー不足の場合、Briefは自分のほとんどをディスクに待避させメモリースペースを開放。◆;や}などに対応した高度なインデントもサポート。◆C言語のテンプレート編集機能で、if文やwhile文等の構文を自動生成。

●安全編集

Briefは、一定時間キーを操作しない場合、その時点の編集内容をファイルに自動保存します。万一、編集中に電源が切れても、最新の保存ファイルから最小の手間で復旧することができます。

◆バッファ書出 切換 削除 リスト表示 ◆ファイル名の指定 表示 変更 ◆バックアップ保存 ◆特殊文字入力 ◆2行を1行に連結 ◆ファイル単位の挿入 ◆増分検索 ◆各種マーク/始・終点の入換 ◆ブロックのカット/コピー/ペースト 印刷 保存 左右へのシフト ◆対応する括弧へ移動 ◆C言語のコメント化 非コメント化 DOSコマンドの実行 繰返し ◆キー操作記憶 ◆入力ヒストリー ◆英文テキスト整形 ◆タブのスペース挿入 ◆拡張子毎に異なった設定ができるタブ ◆検索置換パターンの画面からの入力 ◆英字の大小文字変換 その他

全機種対応パッケージ

5インチ2DDと3.5インチ2DDのFDを一挙に同梱。これ以外のメディアの場合はご請求ください。

定価 48,000円
（消費税は含まず）

日本システム株式会社
●お問い合せは
東京都立川市曙町1-18-2一清ビル立川事業所 Briefユーザー・サポート係
電話 0425-23-4469 FAX 0425-27-6127

BriefはUnder Ware社の商標です。

高価な品物だから、ショップも慎重に選びたい。

No.1 クレジットシステム

1. 9ヶ月先からのお支払いOK
スキップクレジットを御利用になれば支払い開始月を1ヶ月から、最長9ヶ月先までおくらせる事が出来ます。

2. 月々¥1,000からのお支払いOK
月々のお支払い金額の設定が¥1,000からOK。

3. 84回払いもOK
お客様のプランに合わせて、1回から最長84回まで支払い回数をお選びいただけます。

4. ステップアップクレジット
お客様のプランに合わせて、毎月のお支払い金額を徐々に増やしていくシステムです。例えば、1年目は¥3,000、2年目は¥6,000というように、御自由に設定することができます。

5. ボーナス8回払いもOK
毎月の支払いは0（ゼロ）、ボーナス時のみのお支払いでクレジットが御利用になれます。回数は1,2回の他、4・6 ・8回払いまでOK。

6. 業界No.1の低金利

支払い回数	1	3	6	10	12	15	18	20	24	30	36	42	48	54	60	66	72	78	84
他社金利	3	4	5	7	9	10	12	13	16	19	21	25	28	31	35	—	—	—	—
ソフマップ金利	2.0	2.5	3.5	5.0	5.0	8.0	10.0	10.0	10.5	14.5	14.5	18	19	23.5	24.5	30	32	35.5	38.5

7. カードクレジット
各種クレジットカードが店頭だけでなく、通信販売でも御利用になれます。詳しくはお気軽にお問い合わせ下さい。

8. カレッジクレジット
保証人なしで、学生の方でもクレジットが御利用できます。（20歳以上）

No.1 下取りシステム

お持ちの機種を下取りに出して、新品に買替えようと思っている方、ソフマップに御相談下さい。買取り価格がどこよりも高く、新品の販売価格がどこよりも安いから、当然どこよりもお得な条件でお買求めいただけます。又、差額を商品券でお支払いもできます。

No.1 配送システム

到着日指定、夜間配送システム
お客様のご都合に合わせて配送させていただきます。機種によっては、夜間配送できないものがあります。

代金引換システム（要手数料）
係員が品物をお届けに行きますので、その時にお支払い下さい。

No.1 サポートシステム

1. 初期不良交換期間3ヶ月
●万一、お届けした商品が不良の場合、お買い上げ日より3ヶ月以内なら、同等品と即、交換致します。

2. 新品パソコン3年保証
●メーカー保証が1年の場合、メーカー保証1年＋ソフマップ保証2年の計3年間の保証になります。

3. 中古パソコン1年保証
●中古パソコン本体は、1年間保証致します。（ディスプレイプリンタ等は6ヶ月保証となります）

4 新品パソコン買取り保証
●1ヶ月以内であれば必ず買取り保証金額で、下取り、買取り致します。

5. 永久買取り保証
●古くなったパソコン、スクラップ寸前のパソコンでもOK!!どんなパソコンでも、どこよりも高く買い取ります。

ACCEL TOMATO series ICM LAND OMRON 株式会社メルコ maxell I・O DATA SONY FUJI FILM Caravelle WINTECH

新品パソコン・ワープロ・周辺機器・他

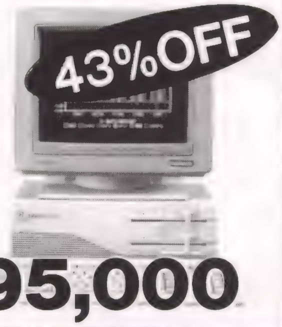
価格は7月31日現在のものです。

JSDMA

●新品パソコン3年保証●新品ワープロ2年保証●初期不良交換期間3ケ月

新品

限定50セット



クレジットも御利用になれます。

●NEC	PC-9801 RX21	¥338,000
●大字(デウ)	HCM-401J (ドットピッチ0.31、カラーCRT)	¥ 84,800
●STAR	BR-2415	¥ 94,800
	標準価格合計	¥517,600

Sofmap特価 ¥295,000

製品	標準価格	特価
36%OFF NEC PC-9801	¥338,000	¥215,000
29%OFF EPSON PC-286VG-STD	¥268,000	¥189,800
EPSON PC-286 BOOK-STD	¥258,000	ソフマップ特価 ¥お電話にて
SHARP X68000 EXPERT II	¥338,000	ソフマップ特価 ¥お電話にて
FM TOWNS モデル2H	¥548,000	ソフマップ特価 ¥お電話にて
TOSHIBA J-3100 SS-002	¥228,000	ソフマップ特価 ¥お電話にて

本体

●NEC	PC-9801RA21	¥498,000	¥315,000
●NEC	PC-9801RS21	¥398,000	¥お電話にて
●NEC	PC-9801TmodelW2	¥548,000	¥お電話にて
●NEC	PC-9801TmodelS5	¥850,000	¥お電話にて
●NEC	PC-9801NS	¥298,000	¥お電話にて
●NEC	PC-9801N	¥248,000	¥165,000
●EPSON	PC-386VR-STD	¥348,000	¥お電話にて
●EPSON	PC-386M-STD	¥328,000	¥お電話にて
●EPSON	PC-286NoteF	¥198,000	¥お電話にて
●TOSHIBA	J-3100SS-02E	¥328,000	¥お電話にて
●TOSHIBA	T-1000(ダイナブック米国版)	¥——	¥お電話にて

モニター

●NEC	PC-KD854N	¥ 84,800	¥46,000
●NEC	PC-KD853N	¥118,000	¥57,000
●NEC	PC-KD862S	¥ 99,800	¥69,800
●NEC	PC-TV454	¥128,300	¥75,500
●NEC	PC-TV455	¥147,000	¥86,000
●NEC	N-5913U	¥120,000	¥85,000
●MITSUBISHI	XC-1498CII	¥107,000	¥お電話にて
●SHARP	CU-14FD	¥ 74,800	¥44,800
●EPSON	CR-4000	¥ 94,800	¥お電話にて
●ACCEL	アクセルビジョン	¥ 99,800	¥43,000
●EPSON	CR-5000	¥138,000	¥お電話にて

プリンター

●NEC	PC-PR201GS	¥135,000	¥96,000
●NEC	PC-PR201J	¥215,000	¥135,000
●NEC	PC-PR201X	¥275,000	¥185,000
●NEC	PC-PR101GS	¥115,000	¥83,000
●NEC	PC-PR101TL3	¥ 69,800	¥46,000
●NEC	PC-PR150VH	¥ 99,800	¥69,000
●NEC	NM-4150	¥245,000	¥129,000
●EPSON	VP-2050PC	¥158,000	¥104,000
●EPSON	AP-850PC	¥ 96,800	¥67,000
●Canon	BJ-130J	¥198,000	¥お電話にて
●Star	CR-3415CL	¥148,000	¥75,000

ワープロ

●NEC	文豪ミニ7RA	¥278,000	¥173,000
●NEC	文豪ミニ5RG	¥148,000	¥105,000
●NEC	文豪ミニ5RD	¥188,000	¥113,000
●TOSHIBA	ルポJW88F	¥165,000	¥お電話にて
●TOSHIBA	ルポJW95FG	¥198,000	¥お電話にて
●TOSHIBA	ルポJW95HD	¥228,000	¥お電話にて
●Canon	キャノワードα50	¥198,000	¥お電話にて
●Panasonic	U1P-603Aix	¥198,000	¥139,000
●Fujitsu	オアシス30MX	¥148,000	¥お電話にて
●Fujitsu	オアシス30LXII	¥188,000	¥お電話にて
●Fujitsu	オアシス30AD	¥158,000	¥お電話にて

ハードディスク

●アイシーエム	SRC-40+I F2750	¥138,000	¥102,000
●アイシーエム	SRC-80+I F2750	¥188,000	¥139,000
●日本テクサ	TRUST40S	¥138,000	¥75,000
●日本テクサ	TRUST80S	¥218,000	¥118,000
●日本テクサ	TRUST40E	¥128,000	¥76,000
●日本テクサ	STATION40S+HA-55S	¥133,000	¥94,000
●ニューテック	NT100-55	¥198,000	¥138,000
●緑電子	PS-40+MDC551	¥127,800	¥93,000
●緑電子	PS-90+MDC551	¥197,800	¥144,000
●スナッパー	PD-40S	¥138,000	¥98,000
●アイテック	IT-SS40	¥138,000	¥82,000

ビジネスソフト

●一太郎Ver.4.3	¥ 58,000	¥36,500
●一太郎ダッシュ	¥ 39,000	¥26,400
●新松	¥ 58,000	¥35,500
●P1 EXE	¥ 48,000	¥29,300
●P1 EXE Plus	¥ 58,000	¥36,000
●花子 Ver.2	¥ 58,000	¥39,800
●CANDY 4	¥ 58,000	¥42,000
●ビジ・コンポ	¥ 40,000	¥25,800
●アシストシリーズ	¥ 9,700	¥ 7,280
●全略ハードディスク殿	¥ 18,000	¥12,000
●メモリープロ386	¥ 10,000	¥ 7,900
●ロータス1.2.3R2.2	¥ 98,000	¥59,800
●マルチプラン4.1	¥ 63,000	¥49,800
●ザ・カード3	¥ 48,000	¥34,800
●サクセスノート	¥ 39,000	¥29,250
●忍者PRO3	¥ 42,000	¥26,800
●クイックベーシック Ver.4.5	¥ 20,000	¥12,800
●ターボC2.0	¥ 29,800	¥22,000
●ロータスフリーランス	¥ 98,000	¥59,800
●まいと～くVer.1.2	¥ 28,000	¥17,900
●ミュージ郎	¥158,000	¥126,400
●ミュージ郎 Jr.	¥ 99,800	¥78,400
●TOPシリーズ	¥——	¥お電話にて
●PCAシリーズ	¥——	¥お電話にて

RAMボード

●メルコ	EMJ-1000mkIII	¥ 27,000	¥18,800
●メルコ	EMJ-2000mkIII	¥ 44,800	¥33,800
●メルコ	EMJ-4000mkIII	¥ 79,800	¥59,800
●メルコ	EXJ-200mkII	¥ 42,000	¥31,900
●メルコ	MRA-2000	¥ 49,800	¥33,800
●IOデータ機器	PIO-PC34H×2M	¥ 45,000	¥33,000
●IOデータ機器	PIO-PC34H×4M	¥ 80,000	¥59,000
●IOデータ機器	PIO-31SS-2MA	¥ 46,000	¥38,800
●メルコ	RCJ-2000	¥ 46,000	¥38,800

ドライブ・周辺機器

●ドライブアクセル	FDC-58	¥ 99,800	¥57,000
●ドライブアクセル	FDC-358	¥ 98,000	¥58,000
●モデム	オムロンMD-1200AIII	¥ 19,800	¥14,900
●モデム	オムロンMD-24FS5	¥ 49,800	¥34,500
●モデム	アイワ PV-A24MNP5	¥ 54,800	¥34,500
●サウンドボード	NEC PC-9801-26K	¥ 25,000	¥19,000
●ハンディースキャナー	オムロン HS-10R2	¥ 49,800	¥34,500
●マウス	NEC PC-9872R	¥ 10,000	¥ 7,980
●マウス	アクセル MS-1	¥	¥ 2,600

●掲載価格は、消費税が含まれておりません。●御注文の前に必ず、商品の有無及び金額を確認して、その上で御送金下さい。

店頭に直接来られる方は ☎東京03-258-3156 ☎大阪06-647-0562

通信販売をご利用の方は ☎東京03-253-4230 ☎大阪06-633-7224
FAX.03-253-4290

札幌 011-865-7030	横浜 045-311-3441	広島 082-222-0604
仙台 022-268-3405	金沢 0762-21-7045	福岡 092-752-0044
新潟 0252-22-6139	名古屋 052-332-2117	高松 0878-34-8833

24時間テレフォンサービス ☎03-258-7910

フリーダイヤル 商品発送のお問合わせ 0120-08-0113

フリーダイヤル 故障・修理のお問合わせ 0120-11-0292

振込先 東京秋葉原店 三和銀行秋葉原支店(普)1012131 口座名義 ㈱ソフマップ

営業時間／年中無休 ●平日 AM11:00～PM8:00 ●日・祭日 AM10:00～PM7:00

株式会社ソフマップ 旧名マップジャパン

〒101 東京都千代田区外神田3丁目15番6号 小暮末広ビル1F
〒556 大阪市浪速区日本橋5丁目7番17号 ソフマップビル

●掲載の商品以外にも多数取り扱いしておりますので、お気軽にお問い合わせ下さい。又、商品在庫は毎日変動しますので品切れの際は御予約承ります。

■NEC・FUJITSU・SHARP・EPSON・SANYO・T O E I・TOSHIBA・MITSUBISHI・Sicair・Canon・ACCEL

Sofmap 冬のボーナス一括払い受付中！ TV

For Computer Communication Age

日本テレビ、TBS、フジテレビ、テレビ朝日、テレビ東京系列でCM放映中！ 直営10店舗

株式会社ソフマップ ●商品の組合せ自由自在 ●月々￥1,000からのお支払いOK！

御予算20万～30万円までのおすすめセット

NEC PC-9801RX21 月々￥1,600から

CPU80286、640KBメモリ、1Mバイト5インチFDD2台内蔵、フロッピーディスクモデル

クレジット注文No.1

- PC-9801RX21〈本体〉……￥Sofmap特価
- PC-KD854N〈14″カラーディスプレイ〉……￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉…￥Sofmap特価

標準価格￥446,800 **￥265,000**

￥1,600×72回	ボーナス ￥20,000×12回
￥3,300×48回	ボーナス ￥20,000×8回
￥6,600×48回	ボーナス なし
￥8,500×36回	ボーナス なし
￥12,300×48回	ボーナス なし

NEC PC-9801RX21 月々￥2,700から

クレジット注文No.2

- PC-9801RX21〈本体〉……￥Sofmap特価
- XC-1498CII〈14″カラーディスプレイ、チルトスタンド〉 ￥Sofmap特価
- PC-9872R〈純正マウス〉……￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉…￥Sofmap特価

標準価格￥479,000 **￥283,000**

￥2,700×84回	ボーナス ￥12,000×14回
￥4,700×84回	ボーナス なし
￥7,100×48回	ボーナス なし
￥10,900×30回	ボーナス なし
￥13,100×24回	ボーナス なし

NEC PC-9801n 月々￥1,500から

CPU V30搭載、ユーザーズメモリ640KB、3.5インチFDD1台内蔵、8階調バックライト付液晶ディスプレイ

クレジット注文No.3

- PC-9801n〈本体〉……￥Sofmap特価
- PC-9801n-19〈プリンタケーブル〉……￥Sofmap特価
- PC-PR101TL3〈熱転写カラープリンタ〉……￥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉…￥Sofmap特価

標準価格￥352,800 **￥232,000**

￥1,500×60回	ボーナス ￥20,000×10回
￥5,800×48回	ボーナス なし
￥7,400×36回	ボーナス なし
￥10,700×24回	ボーナス なし
￥14,200×18回	ボーナス なし

EPSON PC-286VG 月々￥1,900から

CPU80286、16MHZ、NO WAIT、80MB HDD内蔵可能

クレジット注文No.4

- PC-286VG・STD〈本体〉……￥Sofmap特価
- CR-4000〈ドットピッチ0.31、14″カラーノングレア〉 ￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉…￥Sofmap特価

標準価格￥386,800 **￥255,000**

￥1,900×84回	ボーナス ￥14,000×14回
￥5,400×60回	ボーナス なし
￥6,400×48回	ボーナス なし
￥8,200×36回	ボーナス なし
￥11,900×24回	ボーナス なし

EPSON PC-286VG 月々￥1,800から

クレジット注文No.5

- PC-286VG・STD〈本体〉……￥Sofmap特価
- XC-1498CII〈14″カラーディスプレイ、チルトスタンド付〉 ￥Sofmap特価
- MS-50HR〈PCマウス〉……￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……￥Sofmap特価

標準価格￥405,800 **￥259,800**

￥1,800×48回	ボーナス ￥28,000×8回
￥2,500×36回	ボーナス ￥35,000×6回
￥5,300×24回	ボーナス ￥40,000×4回
￥8,300×36回	ボーナス なし
￥15,900×18回	ボーナス なし

EPSON PC-286BOOK 月々￥1,500から

小型軽量2FDD、CPU 80C286、12MHz HDD搭載タイプ

クレジット注文No.6

- PC-286BOOK〈本体〉……￥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉・￥Sofmap特価

標準価格￥286,000 **￥208,000**

￥1,500×24回	ボーナス ￥50,000×4回
￥3,600×84回	ボーナス なし
￥5,300×48回	ボーナス なし
￥9,800×24回	ボーナス なし
￥18,600×12回	ボーナス なし

NEC PC-8801MC model2 月々￥1,600から

1Mバイトタイプ5インチFDD2台内蔵、FM音源6音、リズム6音、デジタルサンプリング機能、CD-ROM標準搭載

クレジット注文No.7

- PC-8801MC model2〈本体〉……￥Sofmap特価
- PC-KD862S〈14″カラーディスプレイ〉……￥Sofmap特価
- ゲームソフト2本〈定価￥9,800以下のお好きなソフト〉・￥Sofmap特価
- マクセルブランクディスケット〈5″2D×10枚〉……￥Sofmap特価

標準価格￥334,400 **￥216,000**

￥1,600×48回	ボーナス ￥23,000×8回
￥3,600×84回	ボーナス なし
￥6,900×36回	ボーナス なし
￥13,300×18回	ボーナス なし
￥19,000×12回	ボーナス なし

TOSHIBA J-3100 SS-002 月々￥3,000から

日本語MS-DOS、ATOK7および5標準装備FLサイドライト液晶

クレジット注文No.11

- J-3100SS-002〈本体〉……￥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉…￥Sofmap特価

標準価格￥256,000 **￥217,000**

￥3,000×84回	ボーナス なし
￥4,400×48回	ボーナス なし
￥5,500×36回	ボーナス なし
￥8,100×24回	ボーナス なし
￥11,900×12回	ボーナス ￥20,000×2回

御予算30万～35万円までのおすすめセット

NEC PC-9801RX21 月々￥2,400から

クレジット注文No.9

- PC-9801RX21〈本体〉……￥Sofmap特価
- XC-1498CII〈14″カラーディスプレイ、チルトスタンド付〉 ￥Sofmap特価
- AP-850PC〈48ドット熱転写漢字カラープリンタ〉……￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……￥Sofmap特価

標準価格￥565,800 **￥343,000**

￥2,400×84回	ボーナス ￥20,000×14回
￥5,900×24回	ボーナス ￥60,000×4回
￥8,600×48回	ボーナス なし
￥11,000×36回	ボーナス なし
￥15,900×24回	ボーナス なし

NEC PC-9801RX21 月々￥2,100から

クレジット注文No.10

- PC-9801RX21〈本体〉……￥Sofmap特価
- PC-KD854N〈14″カラーディスプレイ〉……￥Sofmap特価
- PC-PR150VH〈48ドット熱転写漢字プリンタ〉……￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉・￥Sofmap特価

標準価格￥546,600 **￥338,000**

￥2,100×72回	ボーナス ￥25,000×12回
￥4,300×48回	ボーナス ￥25,000×8回
￥5,600×84回	ボーナス なし
￥7,100×60回	ボーナス なし
￥10,800×36回	ボーナス なし

NEC PC-9801RX21 月々￥1,900から

クレジット注文No.11

- PC-9801RX21〈本体〉……￥Sofmap特価
- HCM-401J〈14″RGBカラーディスプレイ〉……￥Sofmap特価
- PC-PR150VH〈48ドット熱転写漢字プリンタ〉…￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉…￥Sofmap特価

標準価格￥546,600 **￥331,800**

￥1,900×60回	ボーナス ￥30,000×10回
￥3,800×36回	ボーナス ￥40,000×6回
￥5,500×84回	ボーナス なし
￥8,300×48回	ボーナス なし
￥15,300×24回	ボーナス なし

NEC PC-9801NS 月々￥2,100から

32ビットCPU搭載、20Mバイト固定ディスク内臓

クレジット注文No.12

- PC-9801NS〈本体〉……￥Sofmap特価
- PC-9801N-19〈プリンタケーブル〉……￥Sofmap特価
- PC-PR150VH〈48ドット熱転写漢字プリンタ〉……￥Sofmap特価
- PC-9872R〈純正マウス〉……￥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉……￥Sofmap特価

標準価格￥442,800 **￥330,000**

￥2,100×84回	ボーナス ￥21,000×14回
￥3,600×60回	ボーナス ￥20,000×10回
￥5,600×84回	ボーナス なし
￥8,300×48回	ボーナス なし
￥10,500×36回	ボーナス なし

EPSON PC-286VG 月々￥2,000から

クレジット注文No.13

- PC-286VG・STD〈本体〉……￥Sofmap特価
- CR-4000〈ドットピッチ0.31、14″カラーノングレア〉 ￥Sofmap特価
- AP-850PC〈48ドット熱転写漢字カラープリンタ〉 ￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉…￥Sofmap特価

標準価格￥483,600 **￥322,000**

￥2,000×72回	ボーナス ￥24,000×12回
￥4,000×60回	ボーナス ￥18,000×10回
￥6,000×72回	ボーナス なし
￥8,000×48回	ボーナス なし
￥12,300×30回	ボーナス なし

TOSHIBA J-3100 SS 02E 月々￥2,200から

標準1MB、最大3MBのユーザーズメモリ、2MB HDD内臓

クレジット注文No.14

- J-3100SS-02E〈本体〉……￥Sofmap特価
- PWS-5271A〈24ドット熱転写漢字カラープリンタ〉……￥Sofmap特価
- BUSI COMPO〈J-3100用ソフト〉……￥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉……￥Sofmap特価

標準価格￥454,000 **￥332,000**

￥2,200×84回	ボーナス ￥20,000×14回
￥5,500×84回	ボーナス なし
￥8,300×48回	ボーナス なし
￥10,600×36回	ボーナス なし
￥15,300×24回	ボーナス なし

SHARP X68000 EXPERT II 月々￥2,600から

CPU68000、クロック10MHz、メインメモリ2M

クレジット注文No.15

- CZ-603C〈本体〉……￥Sofmap特価
- CZ-605D〈15″カラーディスプレイ〉……￥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉…￥Sofmap特価

標準価格￥477,000 **￥345,000**

￥2,600×84回	ボーナス ￥20,000×14回
￥5,900×24回	ボーナス ￥60,000×4回
￥8,600×48回	ボーナス なし
￥11,000×36回	ボーナス なし
￥15,900×24回	ボーナス なし

FUJITSU FM TOWNS-2Fセット 月々￥1,600から

32ビット386™マイクロプロセッサ搭載、540MB、CD-ROM搭載

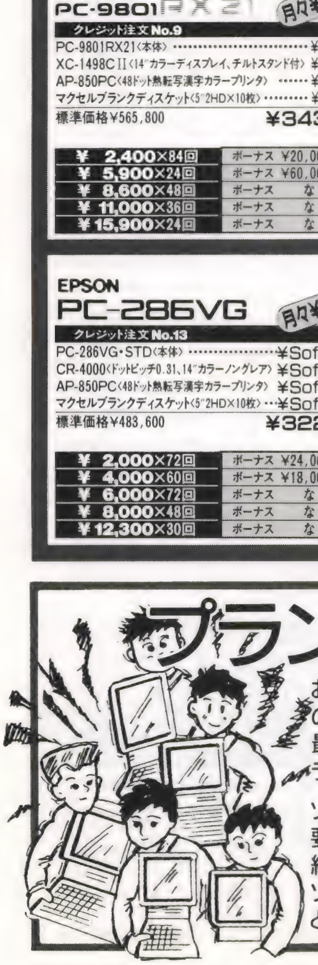
クレジット注文No.16

- FM TOWNS-2F〈本体〉……￥Sofmap特価
- FMT-KB105〈10キー付キーボード〉……￥Sofmap特価
- FMT-DP531〈14″カラーCRT〉……￥Sofmap特価
- B276A010〈システムソフトウェア V1.1〉……￥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉・￥Sofmap特価

標準価格￥517,800 **￥お電話にて**

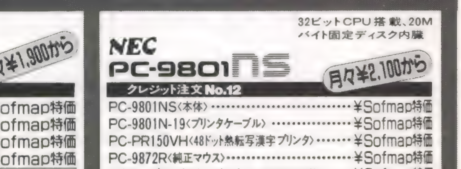
￥1,600×60回	ボーナス ￥30,000×10回
￥3,900×24回	ボーナス ￥65,000×4回
￥5,800×72回	ボーナス なし
￥7,900×48回	ボーナス なし
￥10,200×36回	ボーナス なし

プランに合わせて自由自在！



お客様が10人いらっしゃれば、10通りのパソコンの選び方があっていいのではないか…。価格、機種の特徴、用途etc. 様々なファクターから最も御自身に合ったパソコンをお選びいただくことが、快適なパソコンライフの早道ではないかとソフマップは考えます。

ソフマップクレジットはベーシックな組合せにしてありますので、他に必要なものがあれば自由に追加する事が出来ます。又、お支払い回数や組合せの変更も自由自在ですので、プランに合わせてお選び頂けます。ソフマップはパソコンを通じて、お客様の末永きパートナーになりたいと願っています。



全国出張サービス

富士通FM-TOWNSをお買い上げのお客様に万一故障が生じた場合、には、サービスマンが全国どこへでもお伺い致します。

ACCEL TOMATO series ICM LAND OMRON 株式会社 メルコ maxell I·O DATA SONY FUJI FILM Caravelle WINTECH

9ヶ月後からのお支払いもOK！

●お支払い回数は1回〜84回まで自由自在　●頭金なしクレジット

御予算35万〜40万円までのおすすめセット

NEC PC-9801RX21　月々¥2,200から

クレジット注文No.17

- PC-9801RX21〈本体〉……¥Sofmap特価
- PC-KD854N〈14″カラーディスプレイ〉……¥Sofmap特価
- PC-PR201GS〈130桁、24ドットプリンタ〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥581,800　**¥369,800**

¥ 2,200×72回	ボーナス ¥28,000×12回
¥ 3,500×36回	ボーナス ¥50,000×6回
¥ 7,700×60回	ボーナス なし
¥ 11,800×36回	ボーナス なし
¥ 17.000×24回	ボーナス なし

NEC PC-9801RX21　月々¥2,200から

クレジット注文No.18

- PC-9801RX21〈本体〉……¥Sofmap特価
- PC-KD854N〈14″カラーディスプレイ〉……¥Sofmap特価
- PC-PR150VH〈48ドット熱転写漢字プリンタ〉……¥Sofmap特価
- 一太郎 Ver.4〈日本語ワードプロセッサ〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥604,600　**¥375,000**

¥ 2,200×84回	ボーナス ¥24,000×14回
¥ 4,700×48回	ボーナス ¥28,000×8回
¥ 6,900×72回	ボーナス なし
¥ 9,300×48回	ボーナス なし
¥ 14.400×30回	ボーナス なし

NEC PC-9801RX21　月々¥2,700から

クレジット注文No.19

- PC-9801RX21〈本体〉……¥Sofmap特価
- XC-1498CII〈14″カラーディスプレイ、チルトスタンド付〉¥Sofmap特価
- PC-PR201GS〈130桁、24ドットプリンタ〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥604,000　**¥372,000**

¥ 2,700×72回	ボーナス ¥25,000×12回
¥ 3,500×36回	ボーナス ¥50,000×6回
¥ 6,200×84回	ボーナス なし
¥ 9,300×48回	ボーナス なし
¥ 17.200×24回	ボーナス なし

NEC PC-9801RX21　月々¥2,600から

クレジット注文No.20

- PC-9801RX21〈本体〉……¥Sofmap特価
- HCM-401J〈ドットピッチ0.31、14″カラーCRT〉¥Sofmap特価
- PC-PR101TL3〈熱転写カラープリンタ〉……¥Sofmap特価
- TRUST 40S〈内蔵型40MB HDD〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥644.600　**¥379,800**

¥ 2,600×60回	ボーナス ¥32,000×10回
¥ 4,800×48回	ボーナス ¥28,000×8回
¥ 7,900×60回	ボーナス なし
¥ 9,500×48回	ボーナス なし
¥ 12.100×36回	ボーナス なし

NEC PC-9801RA21　月々¥2,200から

クロック20M、1.6MBメモリ

クレジット注文No.21

- PC-9801RA21〈本体〉……¥Sofmap特価
- PC-KD854N〈14″カラーディスプレイ〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥606,800　**¥365,000**

¥ 2,200×84回	ボーナス ¥23,000×14回
¥ 4,100×48回	ボーナス ¥30,000×8回
¥ 7,600×60回	ボーナス なし
¥ 9,100×48回	ボーナス なし
¥ 11,700×36回	ボーナス なし

EPSON PC-286VG　月々¥2,400から

クレジット注文No.22

- PC-286VG・STD〈本体〉……¥Sofmap特価
- CR-4000〈ドットピッチ0.31、14″カラーノングレア〉……¥Sofmap特価
- VP-2050PC〈24ピンドットマトリクス、漢字カラープリンタ〉¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥544,800　**¥363,000**

¥ 2,400×84回	ボーナス ¥22,000×14回
¥ 5,100×60回	ボーナス ¥15,000×10回
¥ 8,600×36回	ボーナス ¥18,000×6回
¥ 11,600×36回	ボーナス なし
¥ 16.800×24回	ボーナス なし

EPSON PC-286VG　月々¥2,300から

クレジット注文No.23

- PC-286VG・STD〈本体〉……¥Sofmap特価
- XC-1498CII〈14″カラーディスプレイ、チルトスタンド付〉¥Sofmap特価
- VP-2050PC〈24ピンドットマトリクス漢字カラープリンタ〉……¥Sofmap特価
- VP-2000CSFS〈カットシートフィーダ、シングルビン〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥587,000　**¥388,000**

¥ 2,400×84回	ボーナス ¥24,000×14回
¥ 5,100×60回	ボーナス ¥18,000×10回
¥ 7,200×72回	ボーナス なし
¥ 12,400×36回	ボーナス なし
¥ 17.900×24回	ボーナス なし

EPSON PC-386M　月々¥2,300から

3.5インチFDD、ICカードドライブのツインドライブ、A4ファイルサイズ

クレジット注文No.24

- PC-386M-STD〈本体〉……¥Sofmap特価
- XC-1498CII〈14″カラーディスプレイ、チルトスタンド付〉¥Sofmap特価
- AP-850PC〈48ドット熱転写漢字カラープリンタ〉……¥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉……¥Sofmap特価

標準価格¥559,800　**¥383,000**

¥ 2,300×72回	ボーナス ¥29,000×12回
¥ 3,100×60回	ボーナス ¥30,000×10回
¥ 7,200×72回	ボーナス なし
¥ 9,600×48回	ボーナス なし
¥ 12,300×36回	ボーナス なし

御予算40万〜50万円までのおすすめセット

NEC PC-9801RX21　月々¥2,600から

クレジット注文No.25

- PC-9801RX21〈本体〉……¥Sofmap特価
- CU-14FD〈ドットピッチ0.31、14″カラーディスプレイ〉・¥Sofmap特価
- PC-PR150VH〈48ドット熱転写プリンタ〉……¥Sofmap特価
- TRUST-40S〈内蔵型40MB-HDD〉……¥Sofmap特価
- PC-9872R〈純正マウス〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥689,600　**¥425,000**

¥ 2,600×84回	ボーナス ¥27,000×14回
¥ 5,600×48回	ボーナス ¥30,000×8回
¥ 8,600×36回	ボーナス ¥30,000×6回
¥ 13,600×36回	ボーナス なし
¥ 19,600×24回	ボーナス なし

NEC PC-9801RA21　月々¥2,700から

クレジット注文No.26

- PC-9801RA21〈本体〉……¥Sofmap特価
- PC-KD854N〈14″カラーディスプレイ〉……¥Sofmap特価
- PC-PR201GS〈130桁、24ドットプリンタ〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉・¥Sofmap特価

標準価格¥741,800　**¥462,000**

¥ 2,700×84回	ボーナス ¥30,000×14回
¥ 3,200×48回	ボーナス ¥50,000×8回
¥ 7,600×84回	ボーナス なし
¥ 9,600×60回	ボーナス なし
¥ 14,700×36回	ボーナス なし

NEC PC-9801RA21　月々¥3,000から

クレジット注文No.27

- PC-9801RA21〈本体〉……¥Sofmap特価
- XC-1498CII〈14″カラーディスプレイ、チルトスタンド付〉¥Sofmap特価
- PC-PR201GS〈130桁、24ドットプリンタ〉……¥Sofmap特価
- PC-9872R〈純正マウス〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥754,000　**¥480,000**

¥ 3,000×84回	ボーナス ¥30,000×14回
¥ 4,200×60回	ボーナス ¥35,000×10回
¥ 5,300×36回	ボーナス ¥60,000×6回
¥ 8,000×84回	ボーナス なし
¥ 10,000×60回	ボーナス なし

EPSON PC-286VG　月々¥2,600から

クレジット注文No.28

- PC-286VG・STD〈本体〉……¥Sofmap特価
- CR-4000〈ドットピッチ0.31、14″カラーノングレア〉……¥Sofmap特価
- VP-2050PC〈24ピンドットマトリクス、漢字カラープリンタ〉¥Sofmap特価
- DASH40S〈内蔵型40MB-HDD〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥672,800　**¥441,000**

¥ 2,600×72回	ボーナス ¥33,000×12回
¥ 4,300×48回	ボーナス ¥40,000×8回
¥ 6,600×36回	ボーナス ¥45,000×6回
¥ 8,100×72回	ボーナス なし
¥ 11,000×48回	ボーナス なし

EPSON PC-386VR　月々¥2,500から

32ビット、CPU386SX、16MHz、20/40/80バイトのハードディスクを内蔵可能

クレジット注文No.29

- PC-386VR・STD〈本体〉……¥Sofmap特価
- CR-4000〈ドットピッチ0.31、14″カラーノングレア〉……¥Sofmap特価
- VP-2050PC〈24ピンドットマトリクス、漢字カラープリンタ〉¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥624,800　**¥432,000**

¥ 2,500×84回	ボーナス ¥28,000×14回
¥ 4,600×36回	ボーナス ¥55,000×6回
¥ 8,900×60回	ボーナス なし
¥ 10,800×48回	ボーナス なし
¥ 13,800×36回	ボーナス なし

EPSON PC-386M　月々¥2,600から

クレジット注文No.30

- PC-386M・STD〈本体〉……¥Sofmap特価
- CR-4000〈ドットピッチ0.31、14″カラーノングレア〉¥Sofmap特価
- AP-850PC〈48ドット熱転写漢字カラープリンタ〉¥Sofmap特価
- AP-CSF〈カットシートフィーダ〉……¥Sofmap特価
- マクセルブランクディスケット〈3,5″2HD×10枚〉・¥Sofmap特価

標準価格¥567,600　**¥402,000**

¥ 2,500×72回	ボーナス ¥30,000×12回
¥ 3,400×48回	ボーナス ¥40,000×8回
¥ 6,700×84回	ボーナス なし
¥ 8,500×60回	ボーナス なし
¥ 13,000×36回	ボーナス なし

SHARP X68000 EXPERTII-HD　月々¥2,700から

クレジット注文No.31

- CZ-613C〈本体〉……¥Sofmap特価
- CZ-613D〈15″カラーディスプレイ〉……¥Sofmap特価
- マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥607,000　**¥448,000**

¥ 2,700×60回	ボーナス ¥40,000×10回
¥ 4,300×36回	ボーナス ¥60,000×6回
¥ 9,300×60回	ボーナス なし
¥ 14,300×36回	ボーナス なし
¥20,500×24回	ボーナス なし

FUJITSU FM TOWNS-2Hセット　月々¥2,500から

3.5インチFDD2基、ハードディスク40MB内蔵、メインメモリ2MB

クレジット注文No.32

- FM TOWNS-2H〈本体〉……¥Sofmap特価
- FMT-KB105〈10キー付キーボード〉……¥Sofmap特価
- FMT-DP531〈14″カラーCRT〉……¥Sofmap特価
- B276A010〈システムソフトウェア〉……¥Sofmap特価
- マクセルブランクディスケット〈3.5″2HD×10枚〉・・¥Sofmap特価

標準価格¥687,800　**¥お電話にて**

¥ 2,500×84回	ボーナス ¥30,000×14回
¥ 3,600×60回	ボーナス ¥35,000×10回
¥ 4,400×36回	ボーナス ¥60,000×6回
¥ 7,500×84回	ボーナス なし
¥ 9,400×60回	ボーナス なし

●掲載価格は、消費税が含まれておりません。●御注文の前に必ず、商品の有無及び金額を確認して、その上で御送金下さい。

店頭に直接来られる方は　東京03-258-3156　大阪06-647-0562

通信販売をご利用の方は　東京03-253-4230　大阪06-633-7224

FAX.03-253-4290

札幌 011-865-7030　横浜 045-311-3441　広島 082-222-0604
仙台 022-268-3405　金沢 0762-21-7045　福岡 092-752-0044
新潟 0252-22-6139　名古屋 052-332-2117　高松 0878-34-8833

24時間テレフォンサービス　03-258-7910
フリーダイヤル 商品発送のお問合わせ　0120-08-0113
フリーダイヤル 故障・修理のお問合わせ　0120-11-0292

振込先　東京秋葉原店
三和銀行秋葉原支店(普)1012131
口座名義 ㈱ソフマップ

営業時間／年中無休
●平日 AM11：00〜PM8：00
●日・祭日 AM10：00〜PM7：00

株式会社ソフマップ
旧名マップジャパン

〒101　東京都千代田区外神田3丁目15番6号 小暮末広ビル1F
〒556　大阪市浪速区日本橋5丁目7番17号 ソフマップビル

●掲載の商品以外にも多数取り扱いしておりますので、お気軽にお問い合わせ下さい。又、商品在庫は毎日変動しますので品切れの際は御予約承ります。

NEC・FUJITSU・SHARP・EPSON・SANYO・T O E I・TOSHIBA・MITSUBISHI・Star・Canon・ACCEL

Sofmap 月々のお支払いは¥1,000からOK！回数も1回〜84回まで自由

For Computer Communication Age

株式会社ソフマップ ●ボーナス1・2・4・6・8回払いOK ●カレッジクレジット ●スピードクレジット

御予算50万円以上のおすすめセット

NEC PC-9801RA21 月々¥3,000から

クレジット注文No.33

PC-9801RA21〈本体〉……¥Sofmap特価
PC-KD854N〈14″カラーディスプレイ〉……¥Sofmap特価
BJ-130J〈48ドットバブルジェットプリンタ〉……¥Sofmap特価
プリンターケーブル……¥Sofmap特価
PC-9872R〈純正マウス〉……¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉・¥Sofmap特価

標準価格¥820,600 **¥502,000**

¥ 3,000×84回	ボーナス ¥32,000×60回
¥ 3,800×60回	ボーナス ¥40,000×10回
¥ 6,800×36回	ボーナス ¥55,000×4回
¥ 9,300×72回	ボーナス なし
¥12,500×48回	ボーナス なし

NEC PC-9801RX21 月々¥3,300から

クレジット注文No.34

PC-9801RX21〈本体〉……¥Sofmap特価
XC-1498II〈14″カラーディスプレイ、チルトスタンド付〉¥Sofmap特価
PC-PR201J〈日本語シリアルプリンタ〉……¥Sofmap特価
SRC-40〈SCSI対応外付40MBHDD〉……¥Sofmap特価
一太郎Ver.4〈日本語ワードプロセッサ〉……¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉・¥Sofmap特価

標準価格¥880,000 **¥551,000**

¥ 3,300×84回	ボーナス ¥35,000×14回
¥ 4,800×60回	ボーナス ¥40,000×10回
¥ 9,200×36回	ボーナス ¥50,000×6回
¥10,200×72回	ボーナス なし
¥13,700×48回	ボーナス なし

NEC PC-9801RX21 月々¥3,000から

クレジット注文No.35

PC-9801RX21〈本体〉……¥Sofmap特価
PC-KD854N〈14″カラーディスプレイ〉……¥Sofmap特価
PC-PR201X〈日本語シリアルプリンタ〉……¥Sofmap特価
TRUST40S〈内蔵型40MB、HDD〉……¥Sofmap特価
PC-9872R〈純正マウス〉……¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉…¥Sofmap特価

標準価格¥869,800 **¥534,000**

¥ 3,000×84回	ボーナス ¥35,000×14回
¥ 5,000×48回	ボーナス ¥50,000×8回
¥ 7,000×36回	ボーナス ¥60,000×6回
¥ 9,800×72回	ボーナス なし
¥12,300×54回	ボーナス なし

NEC PC-9801RA21 月々¥4,200から

クレジット注文No.36

PC-9801RX21〈本体〉……¥Sofmap特価
XC-1498CII〈14″カラーディスプレイ、チルトスタンド付〉¥Sofmap特価
PC-PR201X〈日本語シリアルプリンタ〉……¥Sofmap特価
PC-PR201X-03〈トラクタフィーダ〉……¥Sofmap特価
SPC-40〈SCSI対応、外付40MB、HDD〉……¥Sofmap特価
MD-24FS5〈2400bpsクラス5モデム〉……¥Sofmap特価
9801-26K〈サウンドボード〉……¥Sofmap特価
PS-98-017-HMW〈MS-DOS、Ver.3.3B〉……¥Sofmap特価
一太郎Ver.4〈日本語ワードプロセッサ〉……¥Sofmap特価
PC-9872R〈純正マウス〉……¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥1,062,800 **¥709,100**

¥ 4,200×84回	ボーナス ¥45,000×14回
¥ 9,300×36回	ボーナス ¥60,000×6回
¥11,700×84回	ボーナス なし
¥17,600×48回	ボーナス なし
¥22,600×36回	ボーナス なし

EPSON PC-386VR 月々¥3,600から

クレジット注文No.37

PC-386VR・STD〈本体〉……¥Sofmap特価
CR-5000〈ドットピッチ0.28、15″カラーCRT〉……¥Sofmap特価
VP-2050PC〈24ピンドットマトリクス、チルトスタンド付〉……¥Sofmap特価
VP-2000CSFS〈カットシートフィーダ、シングルビン〉・¥Sofmap特価
SRC-80〈SCSI対応外付80MB-HDD〉……¥Sofmap特価
PC-286MS2〈エプソン純正マウス〉……¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉……¥Sofmap特価

標準価格¥891,800 **¥655,000**

¥ 3,600×60回	ボーナス ¥60,000×10回
¥ 7,500×36回	ボーナス ¥80,000×6回
¥10,800×84回	ボーナス なし
¥16,200×48回	ボーナス なし
¥20,900×36回	ボーナス なし

Apple Computer Macintosh 月々¥10,200から

クレジット注文No.38

Macintosh IIfx 4MB/FD〈本体〉……¥Sofmap特価
Innerタイプ100MB-HDD〈Tecパーツ〉…¥Sofmap特価
Apple 8.24ビデオカード……¥Sofmap特価
Apple 13″Color RGBモニタ……¥Sofmap特価
Apple キーボード(J)……¥Sofmap特価

標準価格¥1,798,000 **¥1,628,000**

¥10,200×84回	ボーナス ¥100,000×14回
¥14,800×72回	ボーナス ¥90,000×12回
¥16,900×84回	ボーナス ¥60,000×14回
¥23,800×60回	ボーナス ¥60,000×10回
¥26,900×84回	ボーナス なし

Apple Computer Macintosh 月々¥5,200から

クレジット注文No.39

Macintosh IIcx 2MB/FD〈本体〉……¥Sofmap特価
Innerタイプ40MB-HDD〈Tecパーツ〉……¥Sofmap特価
Apple 8ビットビデオカード……¥Sofmap特価
Apple 13″Color RGBモニタ……¥Sofmap特価
Apple キーボード(J)……¥Sofmap特価

標準価格¥952,000 **¥878,000**

¥ 5,200×84回	ボーナス ¥56,000×14回
¥ 8,300×60回	ボーナス ¥60,000×10回
¥ 9,500×84回	ボーナス ¥30,000×14回
¥12,400×60回	ボーナス ¥35,000×10回
¥14,500×84回	ボーナス なし

Version up 下取り差額表

- この表に載っていない商品についてはお電話にてお問合わせ下さい。
- 下取り差額は随時変動します。お電話で確認して下さい。
- 下取り差額のお支払いは、クレジットを御利用になれます。
- 下取り、買取りの査定は無料です。
- パソコン以外の周辺機器、ソフト、書等の買取りも行っています。
- 都内、近郊の方は、出張買取り致します。(有料)

買取り・下取り交換の商品の送り先
〒101 東京都千代田区外神田1-8-3
野水ビル3F ソフマップ
買取りセンター

あなたが今、欲しい機種(新品) 7月31日現在

あなたが今、お持ちの機種	PC-9801 RA21 の場合	PC-9801 RX21 の場合	PC-9801 NS の場合	PC-9801T W2 の場合	PC-286VG STD の場合	PC-286 BOOK の場合	PC-386M STD の場合	J-3100 SS の場合	X68000PRO II (CZ-653C) の場合
下取り機種	交換差額	交換差額	交換差額	交換差額	交換差額	交換差額	交換差額	交換差額	交換差額
PC-9801VX21	¥164,000	¥ 64,000	¥ 85,000	¥230,000	¥ 43,000	¥ 60,000	¥110,000	おつり¥9,000	¥ 55,000
PC-9801VM21	¥184,000	¥ 84,000	¥105,000	¥250,000	¥ 63,000	¥ 80,000	¥130,000	¥ 11,000	¥ 75,000
PC-9801VM2	¥224,000	¥124,000	¥ 40,000	¥290,000	¥103,000	¥120,000	¥170,000	¥ 51,000	¥115,000
PC-9801VF2	¥267,000	¥167,000	¥188,000	¥333,000	¥146,000	¥163,000	¥213,000	¥ 94,000	¥158,000
PC-9801UX21	¥179,000	¥ 79,000	¥100,000	¥245,000	¥ 58,000	¥ 75,000	¥125,000	¥ 6,000	¥ 70,000
PC-9801UV2	¥221,000	¥121,000	¥142,000	¥287,000	¥100,000	¥117,000	¥167,000	¥ 48,000	¥112,000
PC-9801LV21	¥204,000	¥104,000	¥125,000	¥270,000	¥ 83,000	¥100,000	¥150,000	¥ 31,000	¥ 95,000
PC-286V・STD	¥229,000	¥129,000	¥150,000	¥295,000	¥108,000	¥125,000	¥175,000	¥ 56,000	¥120,000
PC-286L・STD・N	¥209,000	¥109,000	¥130,000	¥275,000	¥ 88,000	¥105,000	¥155,000	¥ 36,000	¥100,000
PC-88VA	¥264,000	¥164,000	¥185,000	¥330,000	¥143,000	¥160,000	¥210,000	¥ 91,000	¥155,000
PC-8801MH	¥274,000	¥174,000	¥195,000	¥340,000	¥153,000	¥170,000	¥220,000	¥101,000	¥165,000
CZ-601C	¥199,000	¥ 99,000	¥120,000	¥265,000	¥ 78,000	¥ 95,000	¥145,000	¥ 26,000	¥ 90,000

出張買取りいたします。(都内・近郊の方)

自宅まで係員が品物を取りにうかがいます。(有料)

即、現金にてお支払い

買取 りのみの方も電話お待ちしています。

お送りになる方、又は 直接東京店に来られる方 0120-110-994
直接大阪店に来られる方 06-641-8801

オリジナルビデオプレゼント中！

●ただ今、30万円以上お買い上げのお客様には、パソコンを100％活用するためのノウハウがぎっしり詰まったビデオソフト4種類(PC入門編、Macintosh入門編、デザイナー編、ビジネス編)のうちお好きなソフトをプレゼント／ソフトのみご希望の方は、実費¥2,900にておわけしております。

ACCEL TOMATO series ICM LAND OMRON メルコ maxell I・O DATA SONY FUJI FILM Caravelle WINTECH

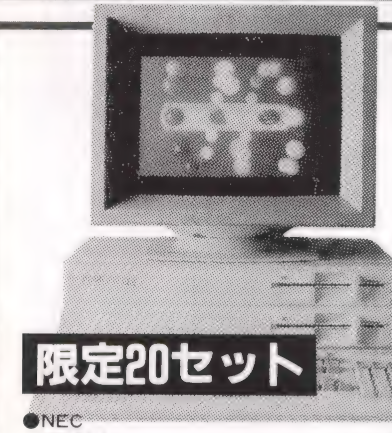
中古パソコン・ワープロTV

日本テレビ、TBS、フジテレビ、テレビ朝日、テレビ東京系列でCM放映中!! 直営10店舗

J-DMA

(このページの掲載商品はすべて中古です)

●中古パソコン1年保証　●代金引換システム



限定20セット

●NEC PC-88VA	¥298,000
●大宇(デウ)HCM-401J (ドットピッチ0.31、カラーCRT)	¥84,800
標準価格合計	¥382,800

Sofmap特価 ¥99,800

本体

■NEC

- ●PC-9801RX21 ····¥338,000➡¥192,800より
- ●PC-9801RA21 ····¥498,000➡¥289,800より
- ●PC-9801RS21 ····¥398,000➡¥228,000より
- ●PC-9801RA2 ·····¥498,000➡¥274,000より
- ●PC-9801VX21 ····¥433,000➡¥174,000より
- ●PC-9801EX2 ·····¥348,000➡¥196,000より
- ●PC-9801UV11 ····¥265,000➡¥142,800より
- ●PC-9801N ·········¥248,000➡¥159,800より
- ●PC-9801LX2 ·····¥448,000➡¥225,000より
- ●PC-9801VM2 ·····¥415,000➡¥112,000より
- ●PC-9801VF2 ·····¥348,000➡¥54,000より
- ●PC-9801U2 ····· ¥298,000➡¥32,800より
- ●PC-8801MC mode12 ··¥199,000➡¥119,000より
- ●PC-8801mkIIFR 30 ····¥178,000➡¥39,800より

■EPSON

- ●PC-286VX・STD ···¥348,000➡¥219,800
- ●PC-286VE・STD ·¥298,000➡¥142,800
- ●PC-286US・STD ···¥248,000➡¥136,000
- ●PC-286LS・STD ···¥478,000➡¥239,800
- ●PC-286LE・STD ···¥368,000➡¥148,000
- ●PC-286L・STD・N ·····¥348,000➡¥119,800
- ●PC-386V・STD ······¥498,000➡¥249,800
- ●PC-386LS・STD ····¥538,000➡¥289,800

■SHARP

- ●X68000PROセット ···¥397,000➡¥229,800 (CZ-652C+CZ－602D)
- ●X68000EXPERTII-HD セット(CZ-613C+CZ・605D) ¥563,000➡¥399,800

■FUJITSU

- ●FM-TOWNSモデル1セット ¥478,000➡¥146,000 (FM-TOWNS1+FMT-DP531)
- ●FM-77AV40EXセット ······¥306,000➡¥71,800 (FM-77AV40EX+FM-TV154)

■TOSHIBA

- ●J-3100SS-001 ·······¥198,000➡¥123,000
- ●J-3100GT-041 ·······¥598,000➡¥389,000
- ●J-3100SGT-041 ·····¥948,000➡¥568,000

- ●Macintosh SE 30FD ·····¥668,000➡¥354,000
- ●Macintosh II 2MFD ·····¥658,000➡¥299,800

モニター

■NEC

- ●PC-KD854N ·········¥84,800➡¥41,800
- ●PC-KD852 ···········¥99,800➡¥37,800
- ●N-5913 ···············¥165,000➡¥61,800

プリンター

■NEC

- ●PC-PR201G ········¥158,000➡¥71,800
- ●NM-9950 ·············¥245,000➡¥75,000

■EPSON

- ●AP-850PC ···········¥96,800➡¥61,800

ワープロ

■FUJITSU

- ●オアシス30LX ········¥198,000➡¥79,800

■TOSHIBA

- ●ルポJW90HX ················➡¥74,800
- ●ルポJW90FII ················➡¥41800

■NEC

- ●文豪ミニ7HG ·········¥198,000➡¥65,000
- ●文豪ミニ5HL ·········¥168,000➡¥44,800

ソフマップインフォメーション

Apple専門ショップ Mac Collection

Macは初めてという方もMac一筋のマニアの方もとにかく一度御来店下さい。

あなたの期待を裏切りません

直接御来店下さる方

東京☎03-5256-2927 大阪☎06-643-7541

COMPUTER PRO SHOP MIDI LAND

ROLAND・YAMAHA・CASIO・KORG等コンピュータミュージックの事なら、すべておまかせ。シーケンサー、サンプリング、ヴォイスエディタ等ハードからソフトまで何でも取りそろえております。SOUND TOOLS実演してます。さわれます。IBM-PCフルサポートOK!

直接御来店下さる方☎03-5256-0477

WORLD SOFMAP 〈シティビル8F〉

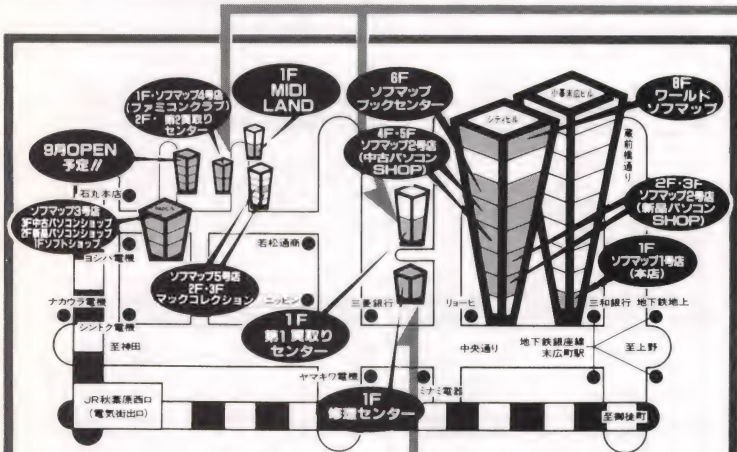
世界のコンピュータ専門ショップ秋葉原に誕生! IBM PC/ATやCommodore AMIGAのソフトが又、今話題のNextが、あなたをお待ちしております。

直接御来店下さる方☎03-5256-2926

BOOK CENTER

パソコンに関する書籍なら、なんでも来いの専門ショップ。コンピュータ関係の本なら言語、OSからファミコン必勝本まで各種取りそろえております。是非一度御来店下さい。

直接御来店下さる方☎03-253-4054



東京秋葉原店

総合買取りセンター

パソコン、ワープロ、FAX、ファミコン、書籍等、何でも買取り致します。

買取りのお問合わせ

☎東京03-5256-2924
☎大阪06-641-8801

下取りのお問合わせ

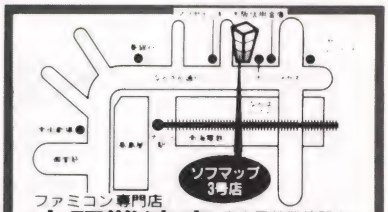
☎東京03-253-4230
☎大阪06-641-8801

営業時間	
●平日	AM11:00～PM8:00
●日・祭日	AM10:00～PM7:00

修理センター

もう安心! ソフマップで買ったものなら、何でも修理致します。

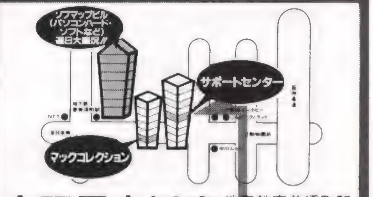
☎東京03-5256-6841
☎大阪06-646-4116



ファミコン専門店

大阪難波店

南海電鉄難波駅より徒歩1分



大阪日本橋店

地下鉄恵美須町駅より徒歩1分

●掲載価格は、消費税が含まれておりません。●御注文の前に必ず、商品の有無及び金額を確認して、その上で御送金下さい。

店頭に直接来られる方は	☎東京03-258-3156 ☎大阪06-647-0562
通信販売をご利用の方は	☎東京03-253-4230 ☎大阪06-633-7224 FAX.03-253-4290 札幌 011-865-7030 横浜 045-311-3441 広島 082-222-0604 仙台 022-268-3405 金沢 0762-21-7045 福岡 092-752-0044 新潟 0252-22-6139 名古屋 052-332-2117 高松 0878-34-8833
24時間テレフォンサービス	☎03-258-7910
フリーダイヤル 商品発送のお問合わせ	0120-08-0113
フリーダイヤル 故障・修理のお問合わせ	0120-11-0292

振込先 東京秋葉原店 三和銀行秋葉原支店(普)1012131 口座名義 (株)ソフマップ

営業時間 / 年中無休 ●平日 AM11:00～PM8:00 ●日・祭日 AM10:00～PM7:00

株式会社ソフマップ 旧名マップジャパン

〒101 東京都千代田区外神田3丁目15番6号 小暮末広ビル1F
〒556 大阪市浪速区日本橋5丁目7番17号 ソフマップビル

●掲載の商品以外にも多数取り扱いしておりますので、お気軽にお問い合わせ下さい。又、商品在庫は毎日変動しますので品切れの際は御予約承ります。

パソコン&AV専門店

キャット

通信販売 受付時間 AM11:00～PM19:00（木曜日定休）

TEL.0482-65-6531

埼玉県川口市芝2-2-7 ㋱はメーカー希望価格です。

NEC PC-9801RA21

㋱¥498,000

¥308,000 消費税 ¥9,240

回数	月々支払い額	ボーナス加算額
24	9,900円	30,000円×4回
36	7,000円	20,000円×6回
48	8,220円	――

NEC PC-9801RX21

㋱¥338,000

¥218,000 消費税 ¥6,540

回数	月々支払い額	ボーナス加算額
24	10,400円	――
36	7,250円	――
48	――	――

NEC PC-9801T W2

㋱¥548,000

¥364,000 消費税 ¥10,920

回数	月々支払い額	ボーナス加算額
24	14,000円	20,000円×4回
36	8,800円	20,000円×6回
48	――	――

EPSON PC-286VG-STD

㋱¥268,000

キャット現金特価

回数	月々支払い額	ボーナス加算額
24	9,250円	――
36	6,450円	――
48	――	――

NEC PC-9801N

㋱¥248,000

¥168,000 消費税 ¥5,040

回数	月々支払い額	ボーナス加算額
24	8,000円	――
36	5,600円	――
48	――	――

NEC PC-9801NS

㋱¥298,000

¥229,800 消費税 ¥6,894

回数	月々支払い額	ボーナス加算額
24	5,880円	30,000円×4回
36	7,580円	――
48	5,980円	――

NEC PC-9801TW5

㋱¥758,000

¥498,000 消費税 ¥14,940

回数	月々支払い額	ボーナス加算額
24	15,500円	50,000円×4回
36	9,980円	40,000円×6回
48	13,100円	――

EPSON PC-286BOOK-STD

㋱¥258,000

間違いなく特価

回数	月々支払い額	ボーナス加算額
24	9,500円	――
36	6,650円	――
48	――	――

シャープ X-68000PROII CZ-653C

㋱¥285,000

間違いなく特価

回数	月々支払い額	ボーナス加算額
24	8,900円	――
36	6,200円	――
48	――	――

シャープ X-68000エキスパートII CZ-613C

㋱¥338,000

間違いなく特価

回数	月々支払い額	ボーナス加算額
24	9,980円	30,000円×4回
36	10,400円	――
48	8,240円	――

富士通 FM-TOWNS2

セット 本体＋システムソフト ディスプレイ キーボード

¥2?8,000

回数	月々支払い額	ボーナス加算額
24	12,300円	――
36	8,580円	――
48	6,700円	――

EPSON PC-386M-STD

㋱¥328,000

間違いなく特価

回数	月々支払い額	ボーナス加算額
24	6,800円	30,000円×4回
36	4,900円	20,000円×6回
48	――	――

東芝 ダイナブック J-3100SS 001

㋱¥198,000

¥132,000 消費税 ¥3,960

回数	月々支払い額	ボーナス加算額
24	6,200円	――
36	4,400円	――
48	――	――

東芝 新型ダイナブック J-3100SS 002

㋱¥268,000

¥199,800 消費税 ¥5,994

回数	月々支払い額	ボーナス加算額
24	9,500円	――
36	6,600円	――
48	――	――

ディスプレイ：プリンター

PC-KD854N	¥46,800
PC-KD853N	¥58,000
PC-TV455	¥84,800
CU-14FD	¥TELにて
CMT-AK142	¥TELにて
PC-PR201GS	¥89,800
PC-PR0101GS	¥79,800
PC-PR150VH	¥68,000

ハードディスク：ドライブ

CRC-MH4B	¥62,000
テクノジャパン TR-40	¥68,800
テクノジャパン TR-80	¥99,800
キャラバル CA-80LG	¥109,800
コニック AW-N100C	¥138,000
コニック AW-N1350CU	¥169,000
SNE.SNE-3(2D)	¥42,000
CRC-FD3.5W	¥43,000

通信販売

お申し込みは簡単、今すぐお電話で!!

1. まずはお電話0482-65-6531へ
2. 商品名、型番を担当へ!! また発送指定日をどうぞ!!
3. 当店指定口座へお振込お願いします。
4. 確認後、商品発送➡お客様の手へ

＊一部の商品に限り納期が2日～10日間かかる時があります。

指定口座／三和銀行わらび支店（普）434123 ㈱キャット宛へ

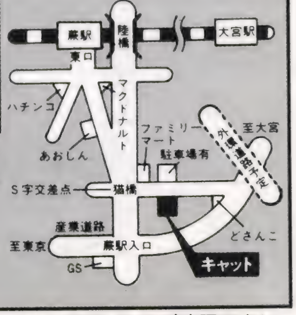
❶ 全品メーカー保証付。
❷ 配送料は1ケ口620～1,620円。（地域により異なります。）
❸ 銀行振り込みはお客様負担です。

若きスタッフ求ム!! 男女正社員

資格
年令18～25才まで、当店まで通える方。固定給15万～

賞与年2回・昇給年1回

業務内容
通販受付・経理・販売・管理。
※定休日・時間面談応ず。

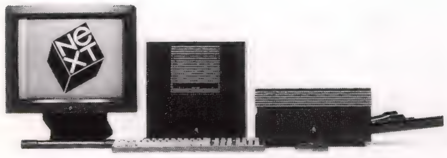


本紙の表示価格には消費税が含まれておりませんので、表示価格＋3%の売価になります。価格は変動することがありますのでTELにてご確認下さい。

ASCII VIDEO

NeXT チュートリアル〈全4巻〉 VIDEO ビデオ

VHS版・4巻化粧箱入り
セット定価18000円(税込)
「限定発売」



NeXTのすべてを見せる!

天才Steven Jobsが
開発したNeXTマシンの特長を、
映像でわかりやすく紹介しています。

第1巻 FIRST IMPRESSION (25分)

- ハードウェア概要
- マウスとウィンドウの操作
- ユーザー登録
- ネットワーク接続

第2巻 OPERATION (21分)

- 基本オペレーション
- アプリケーション操作
- アプリケーション操作の実際

第3巻 PROGRAMMING (29分)

- NeXT流プログラミングとは
- プログラミングの実際
- エピローグ

第4巻 APPLICATION TOUR (25分)

- アプリケーションの紹介
- メールとデジタルライブラリ

制作 キヤノン株式会社
企画協力 株式会社アスキー
発売 株式会社アスキー

御好評につき
追加注文受付中
お申し込み〆切
1990年 8 月末日迄

●お求めは下記の書店へ

■札幌　紀伊国屋書店(札幌店)　旭屋書店(札幌店)■仙台／アイエ書店(駅前店)■浦和／須原屋(本店)■東京　千代田区／書泉グランデ・三省堂書店(本店)　ADOブックショップ　中央区／八重洲ブックセンター・丸善(日本橋店)　港区／雄峰堂書店　渋谷区／大盛堂書店・紀伊国屋書店(渋谷店)　旭屋書店(渋谷店)　新宿区／紀伊国屋書店(本店)　三省堂書店(新宿西口店)　豊島区／芳林堂書店(本店)　旭屋書店(池袋店)　文京区／東大本郷生協■横浜　有隣堂(西口トーヨー店・ルミネ店)栄松堂(ジョイナス店)■川崎／文学堂■厚木／有隣堂(厚木店)■名古屋／三洋堂(パソコンΣ)　三省堂書店(名古屋店)　丸善ブックメイツCP■京都　アバンティブックセンター・ジュンク堂(京都店)■大阪／紀伊国屋書店(梅田店)　旭屋書店(本店)■神戸　ジュンク堂(センター街店)■岡山／紀伊国屋書店(岡山店)■広島／紀伊国屋書店(広島店)■福岡／紀伊国屋書店(福岡店)

または弊社直販部へ直接お申し込みください。**直販部(03)486-7114**

〒107-24 東京都港区南青山6-11-1 スリーエフ南青山ビル ㈱アスキー書籍 雑誌営業部 TEL(03)486-1977 **株式会社アスキー**

資料請求 No.205

アスキーはあなたと一緒に成長する新しいタイプの会社です。

アスキー1991年新卒者募集

アスキーは若い会社です。社長のそして社員の年齢が若いというだけではありません。社員ひとりひとりの柔軟な感性、成長への意欲、個性のぶつかり合いが、アスキーという会社の個性を作りだしている。そして、これこそがすでに完成された企業とは違うアスキーの「若さ」の証明です。アスキーはあなたの個性を伸ばし、あなたはアスキーを育てる。そんな関係を大切にしていきたいと思います。

■募集要項

●募集学科　全学部全学科
●募集職種　技術職(ソフトウェア/ハードウェア)
・UNIXを中心とするOS
・システムソフトウェア
・各種アプリケーションソフトウェア
・アミューズメントソフトウェア
・LSI/各種システムの企画・設計
・MSX関係システムの企画・設計・開発
・各種出版技術(DTP・電子編集等)
・パソコン通信ネットワーク　他
製品企画・マーケティング職(全事業)
営業職(法人営業・販売店営業・ユーザーサポート 他)編集・制作職(主に出版事業)
事務管理職(財務・法務・総務・人事・秘書サポート業務) 他
●募集人員　100名
●応募方法　自由応募
●提出書類　履歴書(写真添付)、卒業見込証明書、成績証明書、健康診断書
●選考方法　作文、面接

■待遇、勤務(1990年度実績)

●初任給　大学院卒　¥226,400
大学卒　¥210,000
短大・専門学校卒　¥188,900
※諸手当含む
●昇　給　年1回(4月)
●賞　与　年2回(6月・12月)
●勤務地　東京(南青山)、川崎(南黒川)
●勤務時間　9:30～17:30
●休　日　完全週休2日制、祝祭日、年末年始
●休　暇　年次有給休暇(初年度12日、翌年以降2日ずつ増加、最大20日まで)
特別休暇(慶弔など)
●福利厚生　制度/社会保険(健康保険・厚生年金・雇用保険・労災保険)
財形預金・財形基金、社員持株制度、住宅ローン利子補給制度、退職年金制度
会員/スポーツクラブ法人会員(青山倶楽部、エグザス)
ラフォーレクラブ健康保険組合会員
施設/独身寮(世田谷・三鷹台のほか91年度都内新規購入予定)
健康保険組合所有各種保養施設
研修/新入社員研修 他

■会社概要

●設　立　1977年5月
●資本金　56億2600万円
●売上高　272億3600万円(1990年3月)
●社員数　630名(1990年6月現在)
●代表者　取締役社長　西 和彦
●事業内容　出版事業
パーソナルソフトウェア事業
アプリケーションソフトウェア事業
システムソフトウェア事業
システム事業
電子出版事業

お問合せは人事部直通フリーダイヤル
0120-246502
0120-246503

受付日時
月曜日～金曜日
10:00 ～ 18:00 (土・日・祝日を除く)

〒107-24 東京都港区南青山6-11-1 スリーエフ南青山ビル ㈱アスキー 人事部　**株式会社アスキー**

資料請求 No.206

ASCII HOT LINE

CANDY4, 3·D PERS発売記念『TAKE OFF』キャンペーン実施中

アスキーセンセーション１　3,000名様にアスキーグッズプレゼント／

アスキーでは、『CANDY4』『3·D PERS（スリー・ディパース）』の発売を記念いたしまして全国有名パソコンショップにて『TAKE OFF』キャンペーンを実施いたしております。期間中（'90年7月1日～'90年9月30日）"クイズ"にお答えいただきますと、毎月1,000名様にアスキー・オリジナルグッズをプレゼントさせていただいております。

詳しくは、〈アスキーサマーキャンペーン'90〉実施店（キャンペーンポスターのある全国有名パソコンショップ）店頭のパンフレットでご覧ください。

アスキーセンセーション２　アスキー・パーソナルCAD大賞 作品大募集／

CANDY4や3·D PERSなど、２次元または３次元パーソナルCADソフトで作成した作品を募集いたします。ソフトウェアの特性を活かしたCAD作品、こんな使い方もあるのかと思われるような独創的な作品を期待いたしております。パーソナルコンピュータを使った製図・デザイン・アートの可能性を、あなたの腕で拡げてください。

●**応募資格**　個人またはグループ。
下記の制作環境であれば、特別な制限はありません。
- ●使用ソフトウェア　CANDY4、3·D PERSなどの市販されている２次元または３次元パーソナルCADソフト
- ●使用ハードウェア　日本語MS-DOSの動作するパーソナルコンピュータ
- ●他のコンテストに未応募作品

●**募集作品**　自由課題です。応募点数の制限はありませんが、未発表のオリジナル作品に限ります。

●**選考基準**　パーソナルCADの特性を最大限利用していると思われるもの。
パーソナルCADの利用法が独創的と思われるもの。

●**審　　査**　第三者立会いのもとに、厳正なる審査を行います。

●**締　　切**　1990年10月31日（当日消印有効）

●**発　　表**　月刊ASCII　1991年1月号（1990年12月18日発売）

●**応募方法**　詳しい応募要項、応募用紙は、〈アスキーサマーキャンペーン'90〉実施店（キャンペーンポスターのある全国有名パソコンショップ）でお配りいたしております。郵送をご希望のお客様は、下記住所までハガキに『アスキー・パーソナルCAD大賞の応募用紙希望』とご明記してご請求ください。
＊応募作品は、入選者と（株）アスキーの間で、データ利用に関する契約を締結させていただきます。なお、応募作品は、返却いたしません。また、応募に関しましては、専用の応募用紙以外での応募は受付できません。ご注意ください。

●**入賞賞品**　全応募作品の中から優秀作品70点には、表彰状と賞品（プリンタ・プロッタなど）を贈呈いたします。

●**協　　賛**　岩崎通信機（株）、エプソン販売（株）、沖電気工業（株）、キヤノン販売（株）、京セラ（株）、グラフテック（株）、シャープ（株）、日本電気（株）、富士ゼロックス（株）、武藤工業（株）、（株）リコー、ローランド ディー・ジー（株）　（50音順）

●**応募用紙請求先、問合わせ先**
〒107-24　東京都港区南青山6-11-1
株式会社アスキー　『アスキーパーソナルCAD大賞』事務局
TEL. 03-486-8080

『アスキーカード』新規会員募集のお知らせ

アスキーカードは、皆様に"充実したパソコンライフ"をお送りいただくためにお届けいたしますクレジットカードです。アスキー製品、ASCII-NETにはもちろん、全国約32万の加盟店を有する日本信販クレジットカードとして、170ヵ国約610万の加盟店を持つVISAカードとしてもご利用いただけるカードです。

●インフォメーション
会員の皆様に新鮮な情報、お得な情報を満載したリフィルサイズの会報誌を毎月発行いたしましてお手元にお届けいたしております。また、パソコン関連のショウ、イベントなどのご案内を随時いたします。

●『体験ディスク』サービス
アスキービジネスソフトの機能、内容を確認してから購入したい。そんな会員の皆様のご希望にお応えいたしまして、機能を制限したシステムとサンプルを収録いたしました『体験ディスク』をご要望に応じまして随時差し上げております。

●アスキーカードパソコン保険
アスキーカード会員の皆様の大切なパソコン本体、周辺機器、ソフトを火災、落雷や盗難、取り扱い上の不注意などさまざまな事故からガードするアスキーカードならではのサービスです。

●ASCII-NET
アスキーのパソコン通信サービス『ASCII-NET』が基本料金のみでご利用いただけます。また、全国の主要都市にアクセスポイントのある『アスキーTri-Pネットワークサービス』もご利用いただけます。

アスキーカードは、18歳以上のお勤め（自営）の方、主婦の方、学生の方（高校生以下はご入会いただけません）にご入会いただけます。

詳しい資料、ご入会をご希望の方は、下記までお電話か、おハガキにてお問い合わせください。折り返し、ご入会の詳しい資料、お申込用紙をお送りさせていただきます。

〒107-24　東京都港区南青山6-11-1
株式会社アスキー　アスキーカード事務局　TEL. 03-797-7601

アスキービジネスパソコンスクール9月度日程

アスキーのビジネスソフトをより活用いただくために、東京・大阪の２会場で下記の日程でパソコンスクールを開催いたしております。

コース名	日　程	料　金
3·D PERS紹介コース	9月11日(火)(10:00-12:00)	無　料
The CARD3入門	9月5日(水)(10:00-17:00)	10,300円(税込)
The CARD3実習	8月22日(水)(10:00-17:00) 9月13日(木)(10:00-17:00)	10,300円(税込)
The CARD3応用	9月20日(木)(14:00-17:00)	5,150円(税込)
The CARD3システム設計	9月26日(水)(10:00-17:00)	20,600円(税込)
Recalc入門	9月7日(金)(14:00-17:00)	5,150円(税込)
パソコン通信入門(東京会場のみ)	8月24日(金)(14:00-17:00)	
CANDY4入門	9月18日(火)(14:00-17:00)	

- ◆各コース共、定員10名となっております。先着順にて受け付けさせていただいておりますので必ずお電話にてご予約ください。
- ◆受講料金のお支払いにアスキーカードをご利用いただきました場合は、受講料金を１割引とさせていただきます。
- ◆詳細につきましては下記教室係までお電話にて直接お問い合わせいただけますようお願い申し上げます。
- ◆アスキービジネスパソコンスクール以外でも、アスキー製品の講習会を受講することができます。東京・大阪地区以外で受講ご希望の方もお気軽にお問い合わせください。

[東京教室]　〒107-24 東京都港区南青山6-11-1 (株)アスキー 教室係 電話03-486-4511
[大阪教室]　〒530 大阪市北区梅田1-11-4 大阪駅前第4ビル (株)アスキー 教室係 電話06-348-0005

アスキーに対するご意見、ご希望、また弊社の販売についてお気付きの点などがございましたら、(株)アスキー営業部「HOT LINE」係宛ハガキにてお送り下さい。よろしくお願いします。

製品についてのお問い合わせは以下の通りです。

＊出版物	486-1977
＊ソフトウェア	486-8080
＊ファミコン	486-5137
＊アスキーネット関連	486-9661

製品のお買い上げ後のご質問はユーザーサポート宛

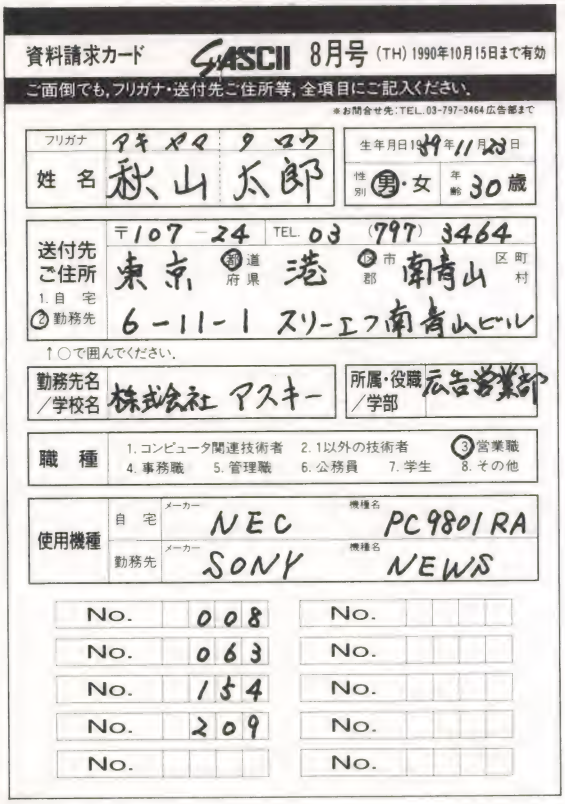
＊出版物＊ゲーム（ファミコンを除く）＊通信ソフト	498-0299
＊ビジネスソフト	498-0205
＊アスキーネット関連	486-9661

〒107-24 東京都港区南青山6-11-1スリーエフ南青山ビル　**株式会社アスキー**

ア ㈱アイ・オー・データ機器 ―― 26
㈱アイザック ―― 表3
㈱エー・エス・ピー ―― 24
エプソン販売㈱ ―― 22,23
カ キャット ―― 204
キャノン㈱ ―― 4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19
クボタコンピュータ㈱ ―― 表4
サ ㈱スタンビジネス ―― 28,29
ソニー㈱ ―― 表2,1
㈱ソフマップ ―― 198,199,200,201,202,203
タ 九十九電機㈱ ―― 27
ナ 日経BP社 ―― 差込み
日本サン・マイクロシステムズ㈱ ―― 20,21
日本システム㈱ ―― 197
ハ 富士ソフトウェア㈱ ―― 193,196
ラ ㈱ライフボート ―― 25,194,195

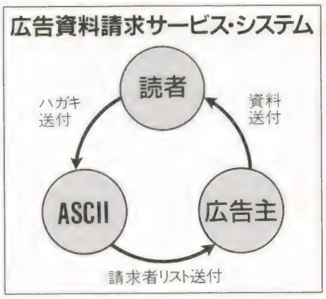
Super ASCIIの 「広告資料請求サービスシステム」

簡単・便利で手間いらず



ぜひ、ご利用ください。

Super ASCIIに掲載された広告製品、サービスに関する資料集めは「広告資料請求サービスシステム」をご利用ください。本誌に添付された資料請求カードをアスキーに返送くださるだけで、ご希望の資料はすべて入手可能です。ぜひ、有効にご活用ください。



●各広告の下にある資料請求番号を「資料請求No.」の欄にご記入ください。

●資料請求カードは、ご面倒でも全項目にご記入ください。

●資料請求カード以外によるご利用は、無効とさせていただきます。

★姓名のフリガナ、お電話番号、ご住所の番地は必ずご記入下さい。記入もれがある場合資料をお届け出来ない場合があります。

お問い合わせ先：〒107-24 東京都港区南青山6-11-1 スリーエフ南青山 株式会社アスキー 広告部　TEL.(03)797-3464

注：フリガナを忘れずお書き下さい。

本誌をご愛読いただきありがとうございます。お近くに書店のない場合は、弊社に直接お申し込み下さい。金額の詳細につきましては書籍広告の頁を参照下さい。

切り取り線

切り取らないで郵便局にお出しください。

この欄は、加入者あての通信にお使いください。注：フリガナを忘れずお書き下さい。

お届け先	お名前		電話	自宅	
				勤務先	
	御住所	〒			

注文書	スーパーアスキー 定期購読	新規申込 年 月号より	継続：定期購読 No.		年間（12冊）8,160円（税込）
	UNIXマガジン 定期購読	新規申込 年 月号より	継続：定期購読 No.		年間（12冊）9,720円（税込）
	MSJ定期購読	新規申込 年 No. より	継続：定期購読 No.		年間（6冊）7,500円（税込）
	品名	円	〒 円	計	円
		円	〒 円	計	円
		円	〒 円	計	円
	合計金額				円
通信欄					

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。（郵政省）

通常払込料金
加入者負担

払込通知票

各票の※印欄は、払込人において記載してください。

口座番号	東京 4－161144	金額	※ 億 千 百 十 万 千 百 十 円
加入者名	株式会社アスキー	料金	払込み / 特殊 円
払込人住所氏名	※ （郵便番号 － ） ご住所 お名前 （☎ － － ）	備考 / 受付局日附印	

記載事項を訂正した場合は、その箇所に訂正印を押してください。

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。（郵政省）

切り取らないで郵便局にお出しください。

通常払込料金
加入者負担

払込票

口座番号	東京 4－161144
加入者名	株式会社アスキー
金額	※ 億 千 百 十 万 千 百 十 円
払込人住所氏名	※
備考	受付局日附印

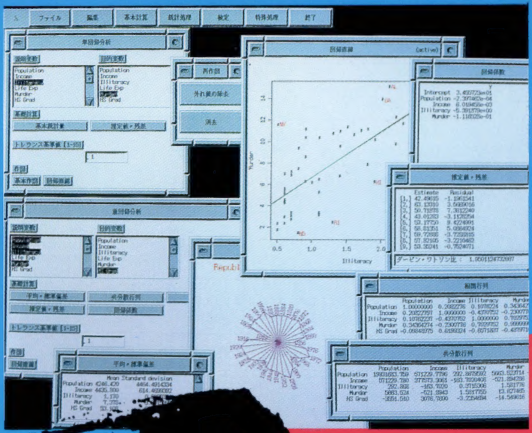
知的マン・マシン・システムを創造する

isac International Sekisui AI Corporation

米国AT&Tのベル研究所で開発され、理想的なデータ解析の環境を実現するSシステム。ウィンドウ・ベースの優れたインターフェイスにより、対話型で高度なデータ解析とグラフィック表示が行えます。また、「言語」と「支援システム」の両方の特徴を兼ね備え、統計調査や財務分析、経営管理、学術研究などのための計算・作図・データ解析など広範囲でご利用いただけます。

《対応機種価格》

ソニーNEWS(X11)……68万円
SUN3(X11)……68万円
SPARC……68万円
HP9000／300……68万円
OMRON／LUNA……68万円
(各機種とも、大学向け特別割引価格……24万円)
(※消費税別)



NEW

'84年リリース以来、1万以上のユーザを獲得してきた統計解析とグラフィックスのための対話型環境、"S"が大幅に機能強化されました。「The New S Language」に基づいた新バージョンの"S"は、以下の特徴を持っています。

- ●S言語仕様が拡張され、オブジェクト指向を取り入れた高水準の関数型言語となっています。これにより、従来のマクロをSの関数として記述することが可能です。
- ●S関数はマクロの4～5倍高速で作動し、記述性も大幅にアップしています。
- ●関数、データ共に、「オブジェクト」として統一的に扱えるようになりました。
- ●UNIXインターフェースをさらに洗練し、C/Fortranで記述した外部関数との接続がより容易になりました。Cの関数中からSの関数を呼び出す事も可能です。
- ●実行時に必要なオブジェクトのダイナミック・ロード機能をサポート。また、倍精度演算、複素数、複素関数を新たにサポートします。
- ●UNIX上の標準ウインドウ・システムであるXバージョン11に対応しウインドウへの多彩なグラフ表示を始め、マウスによるグラフデータのポインティング、ポップアップ・メニューを扱うことができます。
- ●充実したオンライン・ヘルプ機能、ヒストリー機能等、対話性もさらにアップしました。
- ●漢字対応、漢字フォントも標準装備しています。

new S の特徴

①エラーメッセージやヘルプ機能が充実した対話型設計
②豊富なコマンド群を標準装備し、プログラミング不要
③既存のデータベースなどからデータ・コンバートが簡単
④強力なS言語により、独自の解析手法を登録可能
⑤多変量解析や時系列解析などの統計処理用関数を多数装備
⑥解析結果を30種以上の多様なグラフに表示可能
⑦各種プロッタ他、Xウィンドウ、POSTSCRIPTをサポート
⑧解析データの構造や属性等について細かい配慮不要
⑨漢字対応により、漢字等を含むグラフ表示可能

●X Window Systemは、マサチューセッツ工科大学の商標です。
●POSTSCRIPTは、Adobe Systems社の登録商標です。
●UNIXオペレーティング・システムはAT&Tが開発し、ライセンスしています。
●SはAT&Tが開発し、ライセンスしています。

統計解析とグラフィックスのための対話型環境 S

ソフトウェア技術社員募集中。
デモルーム開設中。

株式会社アイザック
〒150 東京都渋谷区渋谷1-11-8 渋谷清水第1ビル
TEL.03-406-1604(営業部直通) FAX.03-406-3910

Super ASCII VOL.1 #2 SEPTEMBER 1990 ISSUE 2 平成２年９月１日発行（毎月１回発行） 第１巻 第２号 通巻２号

資料請求 No.004H

株式会社アスキー 〒107-24 東京都港区南青山6-11-1 スリーエフ南青山ビル
☎(03)486-7111(代表) ☎(03)498-0299(情報電話)
年ぎめ購読料8,160円(税・送料込) 振替 東京4-161144
定価680円(本体660円) Printed in Japan
雑誌05461-9

SUPER ASCII

コンピュータ技術総合誌

9

月刊スーパーアスキー

オペレーティングシステムとは何か

■ToolBook■OS/2 Ver. 2.0

1990

ASCII